Studies in Computational Intelligence 986

Stefka Fidanova *Editor*

Recent Advances in Computational Optimization

Results of the Workshop on Computational Optimization WCO 2020

Springer

Studies in Computational Intelligence

Volume 986

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland

The series "Studies in Computational Intelligence" (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Indexed by SCOPUS, DBLP, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

More information about this series at https://link.springer.com/bookseries/7092

Stefka Fidanova
Editor

Recent Advances in Computational Optimization

Results of the Workshop on Computational Optimization WCO 2020

Springer

Editor
Stefka Fidanova
Institute of Information
and Communication Technology
Bulgarian Academy of Sciences
Sofia, Bulgaria

ISSN 1860-949X ISSN 1860-9503 (electronic)
Studies in Computational Intelligence
ISBN 978-3-030-82396-2 ISBN 978-3-030-82397-9 (eBook)
https://doi.org/10.1007/978-3-030-82397-9

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
This work is subject to copyright. All rights are solely and exclusively licensed 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, expressed 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.

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

Organization

Workshop on Computational Optimization (WCO 2020) is organized in the framework of the Federated Conference on Computer Science and Information Systems FedCSIS—2020.

Conference Co-chairs

for WCO
Stefka Fidanova, IICT-BAS, Bulgaria
Antonio Mucherino, IRISA, Rennes, France
Daniela Zaharie, West University of Timisoara, Romania

Program Committee

Abud, Germano, Universidade Federal de Uberlândia, Brazil
Bonates, Tibérius, Universidade Federal do Ceará, Brazil
Breaban, Mihaela, University of Iasi, Romania
Gruber, Aritanan, Universidade Federal of ABC, Brazil
Hadj Salem, Khadija, University of Tours—LIFAT Laboratory, France
Hosobe, Hiroshi, National Institute of Informatics, Japan
Lavor, Carlile, IMECC-UNICAMP, Campinas, Brazil
Micota, Flavia, West University of Timisoara, Romania
Muscalagiu, Ionel, Politehnica University of Timisoara, Romania
Stoean, Catalin University of Craiova, Romania
Zilinskas, Antanas, Vilnius University, Lithuania

Preface

Many real-world problems arising in engineering, economics, medicine and other domains can be formulated as optimization tasks. Every day we solve optimization problems. Optimization occurs in minimizing time and cost or the maximization of the profit, quality and efficiency. Such problems are frequently characterized by non-convex, non-differentiable, discontinuous, noisy or dynamic objective functions and constraints which ask for adequate computational methods.

This volume is a result of very vivid and fruitful discussions held during the Workshop on Computational Optimization. The participants have agreed that the relevance of the conference topic and quality of the contributions have clearly suggested that a more comprehensive collection of extended contributions devoted to the area would be very welcome and would certainly contribute to a wider exposure and proliferation of the field and ideas.

The volume includes important real problems like modeling of physical processes; workforce planning; parameter settings for controlling different processes, transportation problems and wireless sensor networks; machine scheduling; air pollution modeling; solving multiple integrals and systems of differential equations which describe real processes and solving engineering problems. Some of them can be solved by applying traditional numerical methods, but others need a huge amount of computational resources. Therefore, for them are more appropriate to develop an algorithm based on some metaheuristic methods like evolutionary computation, ant colony optimization, particle swarm optimization, bee colony optimization, constrain programming, etc.

Sofia, Bulgaria
March 2021

Stefka Fidanova
Co-Chair
WCO'2020

Contents

Statistical Measurements of Metaheuristics for Solving Engineering Problems 1
Adis Alihodzic

Heuristic Approaches for the Stochastic Multi-depot Vehicle Routing Problem with Pickup and Delivery 27
Brenner H. O. Rios, Eduardo C. Xavier, Flávio K. Miyazawa, and Pedro Amorim

Evaluation of MO-ACO Algorithms Using a New Fast Inter-Criteria Analysis Method 53
Jean Dezert, Stefka Fidanova, and Albena Tchamova

Semantic Graph Queries on Linked Data in Knowledge Graphs 81
Jens Dörpinghaus and Andreas Stefan

Online Single-Machine Scheduling via Reinforcement Learning 103
Yuanyuan Li, Edoardo Fadda, Daniele Manerba, Mina Roohnavazfar, Roberto Tadei, and Olivier Terzo

Ant Colony Optimization Algorithm for Fuzzy Transport Modelling: InterCriteria Analysis 123
Stefka Fidanova, Olympia Roeva, and Maria Ganzha

Approximation and Exact Algorithms for Multiprocessor Scheduling Problem with Release and Delivery Times 139
Natalia Grigoreva

A Hybrid Method for Scheduling Multiprocessor Tasks on Two Dedicated Processors 157
Méziane Aïder, Fatma Zohra Baatout, and Mhand Hifi

Mathematical Model and Its Optimization to Predict the Parameters of Compressive Strength Test 179
Adeline Goullieux, Mhand Hifi, and Shohre Sadeghsa

Optimal Tree of a Complete Weighted Graph 203
Seyed Soheil Hosseini, Nick Wormald, and Tianhai Tian

Simulation of Diffusion Processes in Bimetallic Nanofilms 221
Vladimir Myasnichenko, Rossen Mikhov, Leoneed Kirilov, Nickolay Sdobnykov, Denis Sokolov, and Stefka Fidanova

On the Problem of Bimetallic Nanostructures Optimization: An Extended Two-Stage Monte Carlo Approach 235
Rossen Mikhov, Vladimir Myasnichenko, Leoneed Kirilov, Nickolay Sdobnyakov, Pavel Matrenin, Denis Sokolov, and Stefka Fidanova

An Analysis on the Degrees of Freedom of Binary Representations for Solutions to Discretizable Distance Geometry Problems 251
Antonio Mucherino

Dynamic Programming for the Synchronization of Energy Production and Consumption Processes 257
Fatiha Bendali, Eloise Mole Kamga, Jean Mailfert, Alain Quilliot, and Helene Toussaint

Reducing the First-Type Error Rate of the Log-Rank Test: Asymptotic Time Complexity Analysis of An Optimized Test's Alternative 281
Lubomír Štěpánek, Filip Habarta, Ivana Malá, and Luboš Marek

Zero Point Approach to Three-Dimensional Intuitionistic Fuzzy Transportation Problem 303
Velichka Traneva and Stoyan Tranev

On Index-Matrix Interpretation of Interval-Valued Intuitionistic Fuzzy Hamiltonian Cycle 329
Velichka Traneva and Stoyan Tranev

On the Conceptual Optimization of Generalized Net Models 349
Velin Andonov, Stoyan Poryazov, and Emiliya Saranova

Sensitivity Study of a Large-Scale Air Pollution Model by Using Optimized Latin Hyprecube Sampling 371
Venelin Todorov, Ivan Dimov, Tzvetan Ostromsky, Zahari Zlatev, Rayna Georgieva, and Stoyan Poryazov

Optimized Quasi-Monte Carlo Methods Based on Van der Corput Sequence for Sensitivity Analysis in Air Pollution Modelling 389
Venelin Todorov, Ivan Dimov, Tzvetan Ostromsky, Zahari Zlatev, Rayna Georgieva, and Stoyan Poryazov

Advanced Stochastic Approaches Based on Lattice Rules for Multiple Integrals in Option Pricing 407
Venelin Todorov

Advanced Stochastic Approaches for Multidimensional Integrals in Neural Networks 425
Venelin Todorov, Stefka Fidanova, Ivan Dimov, Stoyan Poryazov, Stoyan Apostolov, and Daniel Todorov

Improved Stochastic Approaches for Evaluation of the Wigner Kernel 439
Venelin Todorov, Ivan Dimov, and Stoyan Poryazov

A Numerical Study on Optimal Monte Carlo Algorithm for Multidimensional Integrals 451
Venelin Todorov, Stoyan Apostolov, Ivan Dimov, Yuri Dimitrov, Stoyan Poryazov, and Daniel Todorov

Expansions on Quadrature Formulas and Numerical Solutions of Ordinary Differential Equations 463
Venelin Todorov, Yuri Dimitrov, Radan Miryanov, Ivan Dimov, and Stoyan Poryazov

Research of the Use of Battery Shunting Locomotive with Regenerative Brake 477
Tsvetomir Gotsov and Venelin Todorov

Author Index 489

Statistical Measurements of Metaheuristics for Solving Engineering Problems

Adis Alihodzic

Abstract Comparing the results obtained by two or more algorithms in a set of problems is a central task in optimizing or machine learning. Drawing conclusions from these comparisons may require the use of statistical tools such as hypothesis testing. In this paper, we investigate the utilization of parametric multi compared statistical tests on our proposed approach's performance and the rest of the metaheuristics for solving engineering problems. Our proposed strategy (BSGM) includes the Bat algorithm, Simulated annealing, Gaussian distribution, and a novel mutation operator. The proposed method balances the Bat algorithm's critical exploitation and global exploration of the Simulated annealing. The literature's common engineering problems were analyzed in the competition between our BSGM approach and the latest swarm intelligence algorithms. We showed that all algorithms do not produce the same performance using multi compared analysis of variance (ANOVA). The benchmark results also show that our BSGM method provides encouraging results and can compare with the latest metaheuristics according to high-quality solutions and a small number of function evaluations.

Keywords Constrained optimization · Engineering problems · Metaheuristics · Bat algorithm · Simulated annealing · Gaussian distribution

1 Introduction

In the last fifteen years, it was shown that most design nonlinear constrained optimization problems are an essential class of issues in real-world applications, and almost all are characterized as NP-hard problems. For such design optimization problems, finding the best solution may require centuries, even with a supercomputer. These highly nonlinear and multimodal optimization problems are based on the optimiza-

A. Alihodzic (✉)
Department of Mathematics, University of Sarajevo, Zmaja od Bosne 33-35, 71000 Sarajevo, Bosnia and Herzegovina
e-mail: adis.alihodzic@pmf.unsa.ba

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_1

tion of objective functions with complex constraints, usually involving thousands of or even millions of elements. They were written in the form of simple bounds or, more often, as nonlinear inequalities. Nonlinearly constrained optimization problems contain continuous and discrete design variables, nonlinear objective functions, and constraints, some of which may be active at the global optima. Due to the complex nature of an objective function and the constraints that need to be met, it is challenging how to effectively and robustly explore the overall search space. Therefore, practically solving engineering problems are come down to some efficient methods, which are problem-specific [13]. Since optimization methods can not escape falling into some of the local optima, metaheuristics as very modern and efficient global techniques are considered to overcome these type of problems [32]. Besides, those are capable of generating quality solutions in a reasonable amount of time. The creating of quality solutions is related to the establishment of the right balance between exploration and exploitation [30]. Since a magic formula does not exist, which works for all types of problems [34], in this paper, several swarm intelligence algorithms [33] have been adopted for solving nonlinear engineering problems. Some of the most popular swarm intelligence optimization techniques are artificial bee colony (ABC) [3, 5, 20, 29], firefly algorithm (FA) [4, 11, 13, 28], cuckoo search (CS) [14, 23], bat algorithm (BA) [1, 2, 12, 27, 31], flower pollination algorithm [24], and etc. In this article, we have combined the bat algorithm as a representative of swarm intelligent multi-agent algorithm with one agent simulated annealing method to produce as much as possible suboptimal solutions.

The Bat meta-heuristic algorithm (BA) has proposed by Xin-She Yang 2010 [31]. In the paper [2], it has been shown that the BA very well performs a local search, but at times it deviated into some local optima, and it can not reach the optimal solution while solving a challenging problem. The original version of the bat algorithm and the other metaheuristic algorithms were designed to address unconstrained problems. To tackle the constrained problems, bat algorithm (BA) and other algorithms utilize a penalty approach as a constraint handling technique [12]. In this paper, we exploited Deb's rules as a constraint handling process instead of a standard penalty method to improve the solutions' quality. Since the original bat algorithm is not capable of founding the satisfying balance between diversification and intensification, in this paper, we propose an integrated BSGM approach based on simulated annealing (SA) [22], a new mutation operator and Gaussian distribution achieves a right balance and raises overall search performance. The proposed BSGM method was tested on the eight well-chosen benchmark problems, and the descriptive analysis of simulation results tells that our approach almost always wins the state-of-the-art algorithms regarding convergence and accuracy.

Working with stochastic optimization algorithms, statistical comparison plays a vital role in objectively comparing new algorithms to demonstrate their powers and flaws. The statistical estimate of the algorithms of optimization and machine learning has been reported in the papers [7–10, 15, 16, 26]. In this paper, in addition to descriptive analysis, we test hypotheses through statistical parameters to show that the competitive algorithms perform differently in average, i.e. that they do not originate from the same distribution. Also, the existence of a significant statistical difference

between the algorithms supports introducing a new algorithm into the comparative analysis, as in our case, where we compare our BSGM approach and other algorithms in multiple problems. Also, through post hoc analysis, which does a whole range of multiple pair-wise comparisons of algorithms, the significant differences will be located among algorithms' mean performances.

The basic structure of the article looks like this. The basic definitions related to constrained optimization are described in Sect. 2. A brief review of eight engineering optimization problems is there in Sect. 3. The details of our enhanced BSGM approach for solving engineering problems with constraints are in Sect. 4. In Sect. 5, we describe the multi-problem analysis tests by which we evaluate the metaheuristics' performance. Experimental settings are presented in Sect. 6. The results of applying state-of-the-art algorithms for solving engineers problems are presented in Sect. 7. Finally, ending remarks and plan for future work are discussed in the last section of the paper, Sect. 8.

2 Constrained Optimization

The general form of most engineering problems is expressed over objective functions and constraints, usually nonlinear. These problems are considered as constrained optimization problems containing inequality and equality constraints. They become increasingly difficult or even impossible when the traditional techniques are employed for their solving. Generally, their solving can be reduced on the next nonlinear programming problem

$$\min_{\mathbf{x}\in\mathbf{F}\subset\mathbb{R}^n} f(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ is a decision vector composed of n decision variables

$$\mathbf{x} = (x_1, x_2, \ldots, x_n)^T \tag{2}$$

The decision variables x_i may have continuous or discrete values, where each of them is limited by its lower bound L_i and upper bound U_i $(i = 1, \ldots n)$. The objective function f is defined on an n-dimensional hypercube $\mathbf{S}$ such that $S \subset \mathbb{R}^n$. It is used as a measure of effectiveness of a decision. The sets $\mathbf{F} \subseteq \mathbf{S}$ and $\mathbf{U} = \mathbf{S} \setminus \mathbf{F}$ denote feasible and infeasible search space, respectively. The feasible region can being presented as follows

$$\begin{aligned} \psi_k(\mathbf{x}) &\leq 0 \;\; (k = 1 \ldots K) \\ \phi_j(\mathbf{x}) &= 0 \;\; (j = 1 \ldots J), \end{aligned} \tag{3}$$

where letters K and J denote the number of inequality and equality, respectively. If a solution $\mathbf{x} \in F$, then all constraints defined by Eq. 3 must be satisfied. Otherwise,

some of the constraints does not hold. For optimization algorithms, the participation of the equality constraints poses a problem in the sense of reducing available space F, so inequality ones usually replace those in this way

$$|\psi_k(\mathbf{x})| \leq \epsilon \ \ (\forall k), \tag{4}$$

where $\epsilon \geq 0$ is a small violation tolerance.

It is well-known that swarm intelligence algorithms can not directly solve constrained engineering problems because they were designed for unconstrained ones. Therefore, the mapping of constrained problems into unconstrained ones is achieved using a penalty function or utilizing the fly-back mechanism. Using the penalty functions, a constrained issue is being addressed as unconstrained in such a way that infeasible solutions are punished or "penalized" so that the selection process favours feasible solutions. In this way, in the latest phases executing algorithms, the search is directed towards the feasible search space regions. The advantage of penalty functions lies in their simplicity and easy implementation, but the most challenging aspect lies in finding appropriate penalty parameters in pursuit of constrained optimum. Their performance is not always satisfactory, and there is a need for more sophisticated penalty functions.

3 Benchmark Problems

In this part, we will quickly outline eight non-linear design problems to assess our proposed BSGM approach's performance. Each of the eight problems P_i ($i = 1, 2, \ldots, 8$) has discrete and continuous variables. Table 1 summarizes basic characteristics of mentioned problems, such as dimension d, number of linear L_e and non-linear N_e inequalities. Complete mathematical formulation and their description in detail can be found in papers [13, 14].

P1. Pressure Vessel Design Problem

The pressure vessel design problem's basic task is to develop a compressed air room with a pressure of 3×10^3 psi and a minimum volume of 750 ft^3. It is determined as a mixed discrete-continuous constrained problem because it has two discrete variables x_1 and x_2 and two continuous variables x_3 and x_4. These variables have the following meaning: x_1 is a shell thickness, x_2 is a thickness of the spherical head, x_3 is a radius

Table 1 The main properties for the eight benchmark problems

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
d	3	4	3	7	4	2	5	11
L_e	2	2	1	4	0	0	0	0
N_e	1	5	3	7	0	3	1	10

of the cylindrical shell, and x_4 is a shell length. The first two variables x_1, x_2 take the values inside interval [0.0625, 6.1875], while values of the remaining two variables x_3, x_4 belong to interval [10, 200]. The main goal is to decrease the complete charge of the pressure vessel.

P2. Welded Beam Design Problem

The primary objective of the welded beam design problem is to reduce the construction cost of the welded beam subject to restrictions on shear stress τ, bending stress σ in the beam, end deflection δ of the beam and buckling load P_c on the bar. The length of the beam is equal to 14 in, while the force of size 6000 lb is enforced at the end of the shaft. The design variables related to this problem have the following meaning: x_1 is weld thickness h, x_2 present the clamping rod length l, x_3 denotes rod height t, and x_4 is rod thickness b. These variables are bounded by the following limits: $x_1 \in [0.125,\ 5]$, $x_2, x_3, x_4 \in [0.1,\ 10]$.

P3. Tension/Compression Spring Design Problem

The aim of this problem is to reduce the construction cost of the spring, which is limited by four nonlinear constraints. It can be described by three variables x_1, x_2 and x_3, where x_1 is a wire diameter d, x_2 is a mean diameter of the spring D and x_3 is a number of effective coils N. The ranges of those variables are: $x_1 \in [0.05,\ 1.0]$, $x_2 \in [0.25,\ 1.3]$, $x_3 \in [2,\ 15]$.

P4. Speed Reducer Design Problem

The speed deducer design problem is a mixed discrete-continuous optimization problem that describes how to design a simple gearbox. Its application can be exploited between the engine and a light aeroplane propeller to achieve a maximum speed of rotation. The primary goal is to reduce the weight for speed reducer subject to restrictions on bending stress of the gear teeth, surface stress, transverse deflections of the shifts, and stress in the shafts. The variables participating in the construction of speed reducer have the following meaning: x_1 is a face width, x_2 is a module of teeth, x_3 is a number of teeth on the pinion, x_4 and x_5 respectively represent the length of the first and second shaft between the bearings. In contrast, x_6 and x_7 are the first and second shaft diameters, respectively. For these seven variables hold: $2.6 \le x_1 \le 3.6,\ 0.7 \le x_2 \le 0.8,\ 17 \le x_3 \le 28,\ 7.3 \le x_4,\ x_5 \le 8.3,\ 2.9 \le x_6 \le 3.9,\ 5.0 \le x_7 \le 5.5$.

P5. Gear Train Design Problem

The gear train design problem is a discrete optimization problem. It represents a complex issue involving a highly non-linear design space. The determination of volume or centre-to-centre distance of gear is a crucial subject in designing power transmission systems. The gear ratio for a reduction gear train can be defined as the angular velocity ratio between input and output shafts. The total gear train ratio can be defined as follows

$$Gear\ ratio = \frac{w_0}{w_i} = \frac{x_2 x_3}{x_1 x_4} \tag{5}$$

where the variables w_o and w_i present the angular velocities of the output and input shafts, respectively. At the same time, variables x_1, x_2, x_3 and x_4 denote the numbers

of teeth of the gears A, B, C and D, respectively. Those variables take values in the interval [12, 60].

P6. A Truss Design Problem with Three-Bar

The three-bar truss design problem is a continuous optimization problem in civil engineering first proposed by Nowicki in 1974. The purpose of this problem is to seek the optimum cross-section that decreases the weight of the truss. Two design variables x_1 and x_2 are used for its modelling which describe cross-sectional area. The values of mentioned variables are taken from the interval [0, 1].

P7. Cantilever Beam Design Problem

The cantilever beam design problem presents a continuous optimization problem proposed by Fleury and Braibant. It can be described by using five connected square hollow blocks in order to make a beam. The beams are strictly braced at the one end, while a vertical force operates on the cantilever's free end. The main objective of this problem was to minimize the weight of the cantilever. The design space includes five continuous variables x_j and one constraint g_1, where the range of variables x_j $(j = 1, 2, \ldots, 5)$ is the closed interval [0.01, 100.0].

P8. Car Side Impact Design Problem

The car side impact design problem formulated by Gu is a mixed-continuous optimization problem. The overall number of elements in the model is approximately 90000, while the total number of nodes is close to 96000. For side-impact protection, two basic side-impact procedures are NHTSA and EEVC [13]. Based on these procedures, a car was exhibited to a side-impact. The prime goal is to reduce the weight using 11 design variables x_j $(j = 1, \ldots, 11)$ and 10 nonlinear constraints g_k $(k = 1, \ldots, 10)$. The bound conditions for these variables x_j are defined with $0.5 \leq x_j \leq 1.5$ $(j = 1, \ldots, 7), x_8, x_9 \in \{0.192, 0.345\}, -30 \leq x_j \leq 30$ $(j = 10, 11)$.

4 An Enhanced Approach for Solving Constrained Engineering Problems: BSGM

This section provides a hybridized version of the original bat algorithm (BA) proposed by Yang [31] combined with other techniques to solve constrained engineering problems. By analyzing preliminary outcomes shown in the paper [12], we can infer that bat algorithm (BA) has succeeded at least once to produce near-optimal solutions during 30 independent runs. However, although it could generate acceptable solutions using a small number of evaluations, it can be perceived based on experimental results, how it is less stable in contrast to other algorithms. The main disadvantages can be classified as a short seeking of the search space and not a well-established equilibrium between exploitation and exploration. To overcome mentioned drawbacks, we incorporate some parts of the simulated annealing (SA) algorithm as one of the fundamental and often picked heuristic technique [22]. A new mutation operator and

Gaussian perturbations are also integrated into the original bat algorithm to improve its performance. As a result, we provide our BSGM approach to solving engineering optimization problems. By applying this method, the overall stability will be increased because a better exploration disables the algorithm being trapped in some local optimum. Also, as another consequence of that, the enhanced integrated bat algorithm will not iterate until all iterations are exhausted, and it only will require a few iterations for obtaining high-quality solutions. Hence, the proposed BSGM consists of two significant parts similarly as it was done in the case of unconstrained optimization [17]. In the first part, as soon as the algorithm builds the first group of agents, the fittest solutions are changed by novel solutions produced by employing SA, accompanied by the original updating formulas of the bat algorithm. In the second part of the mentioned approach, Gaussian distribution is utilized to scatter locations as much as possible. Also, a new mutation operator was introduced to raise the approach's convergency and establish an acceptable ratio between intensification and diversification. Since all metaheuristic algorithms were initially designed for solving unconstrained problems, in this paper, we require techniques for solving constrained design problems. The use of penalty functions for mapping constrained optimization to an unconstrained one does not commonly deliver satisfactory outcomes because it demands much fine-tuning of the penalty elements that predict the quantity of penalization to be engaged [12]. Thereby, instead of introducing a penalty approach, we decided to employ three of Deb's rules in our BSGM approach. The first Deb's rule tells that an algorithm chooses among two feasible solutions, the one with the better objective function value. Based on the second Deb's rule, a feasible solution beats an infeasible one. In the last Deb's rule, if both solutions are infeasible, the one with the weakest amount of constraint violation was favoured. Some difficulties can appear when the global optima lie on frontier within feasible and infeasible parts. It is essential to highlight here that when we have algorithms that deal with constrained optimization, in the start, they mainly begin with the solutions which are not within the feasible area. Our BSGM approach for constrained problems also does not begin with the feasible initial population. During the running process, Deb's feasibility rules direct the solutions to the feasible region. Hence, slightly infeasible solutions are not discarded but kept in the population. They are utilized in the generation in the next iteration with the hope of giving feasible solutions. In this strategy, initially, larger error values are used, and this value is gradually reduced with each iteration until it reaches whatever acceptable error value. The experimental analysis will show that our proposed BSGM can efficiently perform intensification and diversification of the space compared to the rest algorithms. The details of our proposed BSGM approach are given as follows:

Step 1. Our BSGM method begins by randomly generating population P containing n agents $\mathbf{x}_i = (x_{i,j})_{j=1}^{d}$ $(i = 1, \ldots, n)$ of dimension d, where each vector $\mathbf{x}_i$ can be solution of an engineering problem. Also, in this step are initialized initial loudness A_i, pulse rates r_i and r_i^0 $(\forall i = 1, \ldots, n)$ as well as the annealing constant in SA. Before starting the iterative search process, for each solution, $\mathbf{x}_i$ fitness value is evaluated, and according to Deb's rules, the algorithm identifies both fittest solution

$\mathbf{x}_{best}$ and the smallest violation g_{min}. After that, it determines the starting temperature T_0 and the cycle counter t is reset to 0.

Step 2. Adaptation value of any agent $\mathbf{x}_i$ $(i = 1, \ldots, n)$ in the current temperature t can be depicted as:

$$Av(\mathbf{x}_i) = \frac{e^{-\frac{f(\mathbf{x}_i)-f(\mathbf{x}_{best})}{t}}}{\sum_{i=1}^{n} e^{-\frac{f(\mathbf{x}_i)-f(\mathbf{x}_{best})}{t}}} \tag{6}$$

According to the roulette selection strategy, the alternative solution $\mathbf{x}'_{best}$ was picked up among all bats, while the new formula calculates the new velocity of movement v_i^t

$$\mathbf{v}_i^t = \mathbf{v}_i^{t-1} + (\mathbf{x}'_{best} - \mathbf{x}_i^{t-1}) f_i, \tag{7}$$

where the frequency f_i is being yielded as

$$f_i^t = f_{min} + (f_{max} - f_{min})\beta, \tag{8}$$

and β is a random quantity uniformly extracted from [0, 1], while the letters f_{min} and f_{max} are constants which are usually initialized to 0 and 2, respectively. To additionally boost the heterogeneity of agents into space, we introduced Gaussian operator δ by Eq. 9. Hence, the estimation of the solution $\mathbf{x}_i^t$ is accomplished by driving virtual agents $\mathbf{x}_i^{t-1}$ by the following equation

$$\mathbf{x}_i^t = \delta \mathbf{x}_i^{t-1} + \mathbf{v}_i^t, \tag{9}$$

where $\delta \in N(0, 1)$. In this step, it is necessary to scan the side conditions of the computed new solutions $\mathbf{x}_i^t$.

Step 3. For each solution $\mathbf{x}_i^t$, it should be checked the condition $r_i < rand_i$. If it is satisfied, then the local search is performed around the solution $\mathbf{x}_{best}$ as follows

$$\mathbf{x}_l^t = \mathbf{x}_{best} + a_1 \epsilon, \tag{10}$$

where $a_1 \in (0, 1)$ is a scaling factor, while $\epsilon \in (-1, 1)$ is a random number. As a result, the new solution $\mathbf{x}_l^t$ was generated. Then, as in Step 2, the boundary conditions have to be controlled for each coordinate of the vector $\mathbf{x}_l$. For the experimental purposes, we have fixed the parameter a_1 to value 0.1.

Step 4. In this step, the algorithm performs both computation sum of the violations and fitness value of the selected solution in Step 3. The generated solution from this stage will be accepted as a new one if it is better than the previous one according to Deb's rules or it holds the condition $A_i^t > rand_i$. If one of these two elements is met, then it does perform the update process. It is based on modifying old solutions,

fitness values, and violations with the new ones. Also, in this step, the pulse rate r_i^t was defined by

$$r_i^t = r_i^0(1 - e^{-\beta t}), \tag{11}$$

where $r_i^0 \in$ is an initial pulse rate of the ith agent, and β is a fixed number. The loudness of signal A_i^t was expressed by

$$A_i^t = \alpha A_i^{t-1}, \tag{12}$$

where the changeless factor α behaves likewise to the cooling constant in the SA algorithm. It will be demonstrated throughout the simulation that the most reliable results were found for $r_i^0 = 0.5$, $A_0 = 0.99$, and $\beta = 0.9$.

Step 5. In this step, we apply the new mutation operator $\mathbf{x}_{mut}$ to the previously calculated solutions $\mathbf{x}_i$ to additionally increase the search of the entire scope. The operator $\mathbf{x}_{mut}$ is defined by

$$\mathbf{x}_{mut} = \mathbf{x}_{r_3} + a_2(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}), \tag{13}$$

where r_1, r_2, r_3 are three various randomly chosen numbers in the interval $(0, n)$, and $a_2 \in (0, 2)$ is a scaling factor. Then, we compare the quality of solutions before and after introducing the $\mathbf{x}_{mut}$ operator to attain the optimal solution $\mathbf{x}_{best}$ as well as fitness value $f(\mathbf{x}_{best})$.

Step 6. In this step, we memorize the solution $\mathbf{x}_{best}$ and the highest fitness value according to Deb's rules. Also, the smallest violation g_{min} was determined. The value of temperature parameter T is updated from the cooling schedule, which is defined by

$$T_t = \alpha T_{t-1}, \quad \alpha \in (0, 1), \tag{14}$$

where α is a cooling schedule factor. In our paper, α has values 0.9.

Step 7. The BSGM method stops if the end criterion is reached or the counter t is equal max_no_cycles. In contrast, increment t by one and go to Step 2.

5 Multiple-Problem Analysis Tests

Examining the performance of various algorithms is a fundamental step in many research and practical computational tasks. When new algorithms are proposed, they have to be compared with the state of the art algorithms. Also, when an algorithm is used for a particular problem, its performance has to be compared with others to determine which among them generates the best results. When the differences are evident based on observed results (e.g., when an algorithm is the best in all the problems used in the comparison), the results' direct comparison may be enough. However, this is an unusual situation, and, thus, in most situations, a direct comparison may be

misleading and not enough to draw sound conclusions; in those cases, the statistical assessment of the results is advisable. Working with stochastic optimization algorithms in multiple-problem analysis requires finding a unique representative value from multiple runs for each problem algorithm. For this reason, Garcia et al. [16] suggest using an average of multiple runs as an outstanding value for each algorithm on any problem. The average is an unbiased estimator of the expected value; however, it can be affected by outliers (i.e. low runs of stochastic optimization algorithms) and instead, the median can be used as a representative value. Using the typical approach, either average or median from the multiple runs obtained on a single problem can be used as a representative value involved in the multiple-problem scenario for a specific algorithm on the specific problem. Further, the data obtained for multiple-problem analysis should be analyzed using an appropriate omnibus statistical test.

In many papers, statistical analysis was used as a robust instrument to assess algorithms' achievement and quantify the correlation between algorithm performance and extra factors explaining query properties. In the literature, we discover several techniques for such things: the analysis of variance (ANOVA), t-test, F-test, and least-squares regression, as well as powerful alternatives such as Friedman's test and L1-regression [18]. In order to fairly measure the performance of optimization algorithms in the statistical sense, in this paper, we test null-hypothesis (H_0) and the alternative hypothesis (H_A):

- H_0: there is no statistical significance between the mean performance of the compared algorithms using a set of benchmark problems;
- H_A: there is a statistical significance between the mean performance of the compared algorithms using a set of benchmark problems.

In other words, the null hypothesis claims that each metaheuristic employed to solve engineering problems generates the same performance on average. In contrast, the alternative hypothesis states that there are differences between them that are practically significant, i.e. all metaheuristics do not produce the same mean performance.

In the multiproblem analysis, a value for each pair of algorithm/problem is needed. Also, a multi compared strategy is wanted when there are more than two groups. Multiple comparisons of different techniques must be utilized using a statistical tool to check the related samples' differences. For these purposes, we could use a parametric and nonparametric test. Parametric tests have been generally accepted in the analysis of experiments in data science, and they produce accurate predictions compared to the nonparametric tests. In this paper, we exploit the parametric one-way analysis of variance (ANOVA), which analyses several units' medians to conclude whether they start from the equivalent distribution. In an ANOVA test, the aim is to examine the relationship related to data variability between groups and within groups. At this analysis, the decomposition and assessment of variability are conditioned by various factors. The decomposition of variability in terms of the dependent variable is performed based on the following relation

$$SS = SS_b + SS_w \tag{15}$$

where SS_b and SS_w denote the sums of squared deviations between groups and within groups, respectively. They are defined as follows:

$$SS_b = \sum_{j=1}^{k} n_j(\bar{y}_j - \bar{y})^2 \tag{16}$$

$$SS_w = \sum_{j=1}^{k} \sum_{i=1}^{n_j} (y_{ij} - \bar{y})^2 \tag{17}$$

In the relations (16) and (17), a label k denotes the number of groups, n_j presents the number of elements at the jth group, $\bar{y}_j$ is the mean of the jth group, $\bar{y}$ is the mean of all groups, while y_{ij} is the ith value of the dependent variable at jth group. Similar to the relation (15), the decomposition of the degree of freedom df is performed as follows:

$$df = df_b + df_w \tag{18}$$

where df_b is the degree of freedom between groups and df_w is the degree of freedom within the group. The mean square (MS), the estimation of variance (S^2) and the empirical F-statistic are defined based on the following two relations

$$MS = S^2 = \frac{SS}{df} \tag{19}$$

$$F = \frac{\frac{SS_b}{df_b}}{\frac{SS_w}{df_w}} = \frac{MS_b}{MS_w} = \frac{S_b^2}{S_w^2} \tag{20}$$

To reject the null hypothesis $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k = \mu$, it is desirable to valid $SS_b >> SS_w$, i.e., that the value SS_b be significantly greater than the value SS_w. A probability that a random variable of F-distribution with $df_1 = k - 1$ degree of freedom in the numerator and $df_2 = n - k$ degree of freedom in the denominator ($n = n_1 + n_2 + \cdots + n_k$) exceeds the critical value $F_\alpha(df_1, df_2)$ is called p -value and it speaks about the level of significance of the test. Formally, it can be written as

$$P(F \geq F_\alpha(r_1, r_2)) = \int_{F_\alpha(r_1,r_2)}^{\infty} \frac{\Gamma[(r_1 + r_2)/2](r_1/r_2)^{r_1/2} w^{r_1/2-1}}{\Gamma(r_1/2)\Gamma(r_2/2)(1 + r_1 w/r_2)^{(r_1+r_2)/2}} dw \tag{21}$$

where r_1 and r_2 are the numerator and denominator degrees of freedom, respectively. If p-value ($P(F \geq F_\alpha(r_1, r_2))$) is less than α, the null-hypothesis should be rejected, where α is a theoretical level of significance and it usually takes the value from the set $\{0.05, 0.01\}$. In hypothesis testing through one-factor analysis of variance for an arbitrary statistical variable Y (dependent variable), we compare average values for several groups (more precisely all six algorithms) or more than two levels of the independent categorical variable X.

To apply the ANOVA test, it is necessary to check whether all statistical indices and all groups of algorithms (all levels of the category independent variable X) hold assumptions such as population normality and homogeneity of variances. The normality of the data can be realized in several ways. This section shows how normality is tested by using the Shapiro–Wilk test [25]. The Shapiro–Wilk test calculates a W statistic that tests whether a random sample $x_1, x_2, \ldots, x_n$ comes from (specifically) a normal distribution. The W statistic is calculated as follows:

$$W = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{22}$$

where the $x_{(i)}$ are the ordered sample values ($x_{(1)}$ is the smallest) and the a_i are constants generated from the means, variances and covariances of the order statistics of a sample of dimension n from a normal distribution. The null-hypothesis of this test is that the population is normally distributed. The test outcome is stored as a p-value, which is being compared to the critical value α of the test. Thus, if the p-value is less than the chosen α-level, then the null hypothesis H_0 is rejected, and there is evidence that the data tested are not normally distributed. Otherwise, if the p-value is greater than the chosen α-level, then the null hypothesis can not be rejected, and data come from a normal distribution. For example, if $\alpha = 0.05$, and a p-value of a data set is greater than α critical value, then the Shapiro–Wilk test retains hypothesis H_0, which implies that data belong to a normally distributed population.

It is clear if sample sizes are tiny, for instance, $N < 20$, in that case, it is complicated to meet the normality of the population, i.e. many normality tests typically have low ability to make quality predictions. Due to this and the fact that tiny groups of equal sizes are considered in this paper (each group contains eight samples), more emphasis is placed on the fulfilment of the second assumption of homogeneity of variance, which we test by Levene's statistic [6]. In general, Levene's test is used before running a test like One-Way ANOVA to check the equality of variances over groups or samples. The Levene test can be used to verify that assumption when data comes from a non-normal distribution. Given a variable Y with a sample of size N divided into k subgroups, where N_i is the sample size of the ith subgroup, the Levene test statistic is defined as

$$W = \frac{(N-k)\sum_{i=1}^{k} N_i(\bar{Z}_i - \bar{Z})}{(k-1)\sum_{i=1}^{k}\sum_{j=1}^{N_i}(Z_{ij} - \bar{Z}_i)^2} \tag{23}$$

where $Z_{ij} = |Y_{ij} - \bar{Y}_i|$, and $\bar{Y}_i$ can be the mean, median or the 10% trimmed mean of the ith subgroup. Also, $\bar{Z}_i$ are the group means of the Z_{ij}, while $\bar{Z}$ is the overall means of the Z_{ij}. The three choices for defining Z_{ij} determine the robustness and power of Levene's test. The test's robustness implies its ability to not falsely detect unequal variances when both underlying data are not normally distributed and the variables are indeed equal. The test's power is reflected through its ability to locate unequal variances as soon as the variances are not equal. The null hypothesis H_0 of

Levene's test states that the variances are equal across all samples. The test results are reported as a p-value, which is being compared to the α level of the test. If the p-value is larger than the α level, then the variances are equal, and the hypothesis H_0 is retained; otherwise, the variances are unequal, and the null hypothesis has rejected. Also, the Levene test rejects the hypothesis H_0, if $W > F_{\alpha,k-1,N-k} = F_\alpha$, where $F_{1-\alpha}$ is the lower critical value, and F_α is the upper critical value of the F distribution with $k-1$ and $N-k$ degrees of freedom at a significance level of α.

Based on the ANOVA test, we check whether the null hypothesis H_0 can be rejected. Once the ANOVA test rejects the equivalence of the mean performance of algorithms, detecting differences among them can be done by appealing a post hoc statistical process. This process propagates a whole range of pair-wise comparisons to compare statistically significant differences between algorithms in a standard set of issues [16]. In this paper, we use a very well-known Scheffe post hoc test as very powerful in rejecting the null hypothesis, which identifies pairs of metaheuristics with significant different performances [21]. Scheffe's procedure determines the critical values to reject the null hypothesis as follows:

$$F_S = \frac{(\bar{y}_i - \bar{y}_j)^2}{S_w^2(\frac{1}{n_i} + \frac{1}{n_j})} \tag{24}$$

According to Scheffe's test, we have that the null hypothesis H_0 is being rejected if the F_S statistic value is greater than the Scheffe critical value $df_b \cdot F_\alpha(df_b, df_w)$. Also, the Scheffe test corrects α for simple and complex mean comparisons. The complex mean comparisons involve comparing more than one pair of means simultaneously.

6 Experimental Settings

This section will show through experimentally simulations how our BSGM approach behaves in comparison to well-known algorithms during solving benchmark problems. In the experimental analysis, we have chosen eight well-known constrained engineering problems to direct competition between our approach BSGM and practical algorithms such as SA, ABC, FA, CS, and BA. The experiment's focus is on descriptive analysis for performance algorithm measuring and hypothesis testing, which confirms that there are meaningful differences in statistical sense between the mean performances of competitive algorithms. All algorithms participating in the simulation were carried out on the local machine, which has the following performance:

- **Operating System:** Windows 10×64;
- **Type of processor:** Intel Core i7 3770 K processor with a speed of 3.5 GHz;
- **Memory (RAM)**: 16 GB;
- **Programming language:** C#;
- **Software:** Visual Studio 2019.

6.1 Parameter Settings

As metaheuristics have stochastic properties, each experiment was done in 30 series for each of the problems $P_1, P_2, \ldots, P_8$. The run of each algorithm is over when all its iterations are being consumed. For experimental purposes, each algorithm allocates 2000 iterations. In this case analysis, except for standard control parameters, each of the algorithms has extra control parameters that directly impact the obtained solutions' quality. The adjustments of algorithm parameters are given below:

- **SA**: The temperature T_0 at the beginning is set to 1.0, the stopping temperature T_{stop} was initialized to 1.0E-10, the beginning search period was set to 500, the annealing constant is equal to 0.5, the maximum number of rejections, acceptance and runs are set to 250, 150, and 50, respectively.
- **ABC**: The max. size of population $SP = 40$, the constant 'limit' is initialized to $SP \times d \times 5$, where d denotes the number of variables of the problem, while the modification rate MR and scout production period SPP were set to 0.9 and 400, respectively.
- **FA**: The max. size of firefly population is 40, the initial value of attractiveness β was set to 0.05, the randomization parameter α takes value from [0, 1]. Other parameters were set as $\beta_0 = 1$ and $\gamma = 1.0$;
- **CS**: The maximum population size SP is equal 40 for all benchmark problems. The parameter p_a of catching a cuckoo egg was set to 0.99;
- **BA**: The max. number of agents is 40, the initial values of the pulse rates and loudness are set to 0.5 and 0.99, respectively, the frequencies f_{min} and f_{max} respectively are set to 0 and 2.0, while both constants α and γ are initialized to 0.9;
- **BSGM**: The size of bat population is 40, $f_{min} = 0$, $f_{max} = 2.0, \alpha = 0.9, \gamma = 0.99$, the values of parameters r_i^0 and loudness A_i^0 were initialized to 0.5 and 0.99, respectively. The annealing constant is fixed to 0.5.

7 Results

In this part, we do the experimental results of the algorithms which participate in the competition. In the first part of the simulation results, we perform a formal statistical analysis, while in the second part, we perform a multi compared analysis of variance (ANOVA) to show a statistically significant difference between metaheuristics indeed.

The simulation results observed through descriptive statistics are stored in Table 2. As we can see, for the problem P_1, only the BSGM approach has achieved the best scores for all statistical indices except for the average time, where the BA is two times faster than BSGM. Also, for the same problem, the ABC algorithm produced slightly worse results than the BSGM method, and at the same time, better outcomes than the rest algorithms. Again, for the problem P_2, only the BSGM method has found

Table 2 Descriptive analysis of the obtained results by 30 independent runs

Pr.	Stat.	SA	ABC	FA	CS	BA	BSGM
P_1	*Best*	6099.738697	6059.712773	6059.712977	6059.7184703	6059.796804	6059.712773
	Mean	75604.673833	6059.713833	6059.713518	6059.711971	6079.702933	6059.712773
	SD	4.54E+02	2.66E-06	3.24E-08	7.07E-09	1.18E+01	6.27E-12
	ANI	92649.3	1000	1000	2000	600	1000
	SP	1	25	40	13	40	24
	AT	0.6	0.42	5.91	1.77	0.22	0.41
P_2	*Best*	1.728322	1.724852	1.724852	1.724852	1.725381	1.704852
	Mean	1.739173	1.724874	1.724863	1.724852	2.255381	1.704852
	SD	5.92E-03	6.05E-06	4.02E-05	3.76E-12	5.85E-01	3.65E-13
	ANI	78545.8	1000	500	2000	1000	800
	SP	1	30	30	10	40	26
	AT	0.9	0.79	3.96	1.42	0.54	0.64
P_3	*Best*	0.012708	0.012666	0.012665	0.012666	0.012668	0.011215
	Mean	0.013009	0.012797	0.012694	0.012695	0.019492	0.011215
	SD	2.81E-04	9.96E-05	2.17E-05	2.89E-05	6.55E-03	1.20E-12
	ANI	252033.93	1000	600	1000	2000	684.90
	SP	1	25	30	20	40	22
	AT	1.47	0.48	2.8	1.63	0.65	0.28
P_4	*Best*	2994.842065	2993.542819	2993.542821	2993.542819	2993.668899	2993.542819
	Mean	2998.837062	2993.542819	2993.544598	2993.542819	3007.132712	2993.542825
	SD	2.53E+00	2.48E-12	5.50E-03	2.65E-12	6.53E+00	8.16E-06
	ANI	82264.43	1000	1000	2000	2000	1000
	SP	1	12	40	7	40	38
	AT	1.25	0.25	13.58	1.15	0.95	1

(continued)

Table 2 (continued)

Pr.	Stat.	SA	ABC	FA	CS	BA	BSGM
P_5	*Best*	3.38E-14	2.79E-13	2.55E-20	1.51E-15	5.71E-15	1.11E-31
	Mean	7.19E-10	1.60E-09	2.38E-13	2.79E-09	7.94E-09	2.03E-16
	SD	8.00E-10	3.38E-09	1.20E-12	3.24E-09	4.27E-08	8.58E-16
	ANI	163655.17	60	60	60	50	60
	SP	1	10	10	10	30	10
	AT	0.65	0.01	0.03	0.04	0.01	0.01
P_6	*Best*	263.896818	263.895844	263.895843	263.895844	263.895891	263.852843
	Mean	263.918269	263.895913	263.895843	263.895875	263.907303	263.852843
	SD	1.95E-02	7.28E-05	5.31E-09	2.85E-05	1.63E-02	5.189E-14
	ANI	87690.93	1000	500	2000	1000	926.63
	SP	1	40	35	10	40	17
	AT	0.29	0.35	1.47	0.99	0.19	0.18
P_7	*Best*	1.343702	1.339912	1.339911	1.339912	1.339919	1.339911
	Mean	1.349069	1.339914	1.339911	1.339916	1.433529	1.339911
	SD	3.65E-03	1.26E-06	2.73E-08	2.18E-06	2.80E-01	5.44E-16
	ANI	74087.9	2000	900	2000	1000	811.07
	SP	1	19	40	19	40	24
	AT	0.81	1.01	8.92	3.2	0.53	0.59
P_8	*Best*	22.851120	22.843581	22.842969	22.842824	22.846273	22.842824
	Mean	22.880749	22.855576	22.843086	22.844898	24.010514	22.843767
	SD	7.74E-02	1.57E-02	2.20E-04	1.20E-03	5.92E-01	1.47E-03
	ANI	70471.57	1000	600	2000	1500	842.63
	SP	1	40	40	14	40	35
	AT	0.57	0.98	5.24	2.47	0.82	1.01

ANI Average number of iterations, *SP* Size of population, *AT* Average time (in sec.)

the best optimum, and in the term of statistical parameter S.D., it is a very stable technique. Other algorithms could not reach the optimum, although the ABC, FA, and CS algorithms gave good results in terms of all statistical parameters. Further, the BSGM method has achieved the best result for the problem P_3 as well as the best statistical values, such as mean value and standard deviation. Also, for this problem, the proposed BSGM has consumed the least number of evaluations to generate the best optimum solution. By analyzing the outcomes in Table 2, it can be seen that only BSGM, CS and ABC have delivered the best optimum for the problem P_4, while the FA has generated slightly worse best results. Moreover, the ABC algorithm has used up the smallest number of evaluations and the shortest average time. Since the problem P_5 is not a very hard problem, all algorithms were generated the best optimum. The least number of evaluations have consumed the algorithms ABC, BA and BSGM. Our proposed BSGM has produced the most precise and more stable solutions. For the problem P_6, the BSGM returns both the smallest number of evaluations and the least average time to build the fittest solution. Further, the FA has generated worse best solution than BSGM but slightly better than other algorithms using only 17.500 evaluations. Therefore, for this problem, regarding convergence speed and robustness, the remaining algorithms are considerably inferior to the BSGM algorithm. Considering the outcomes of the P_7, we can observe that the BSGM and FA algorithms were made the best results, where the BSGM has drastically better statistical values such as mean value and standard deviation than the FA. Also, for this problem, equally good results are obtained by ABC and CS algorithms. As it can be seen from Table 2, the achieving global optima have cost 19.465, 68 evaluations by the proposed BSGM, which is almost twice fewer evaluations compared to rest techniques. Finally, by analyzing the experiment's outcomes for the last problem, P_8, we can conclude that only BSGM and CS algorithms achieve the best solution. Also, both FA and ABC algorithms produced acceptable solutions. The descriptive statistics presented in Table 2 confirm that the proposed BSGM approach can accomplish the best solutions from literature for engineering problems $P_1, P_2, \ldots, P_8$. The proposed BSGM works better than the other algorithms concerning the quality and robustness with a noticeably enhanced convergence rate for the bulk of design problems.

The second part of the experiment is based on applying the statistical method for analysis of variance (ANOVA) to confirm hypotheses related to testing differences in the mean performance of the algorithms during solving benchmark problems. For each statistical characteristic or index, we test the null hypothesis. As the strength of the comparison of algorithms depends on the number of parameters they are compared, we will introduce an additional statistical index called ALL, which is obtained as the sum of the previous indices. Thus, we establish six hypotheses related to measuring the mean performance of the algorithms such as SA, ABC, FA, CS, BA and BSGM. In other words, the goal of the hypotheses is to examine whether the algorithms are drawn from the same distribution when compared against statistical indices. In this paper, the statistical indices are BEST, MEAN, S.D., ANI * SP, MEAN TIME, ALL. We point out that when there is a significant statistical difference between the measured algorithms, then the comparison between existing algorithms and new ones is completely justified, which is presented in this paper, in which the

new BSGM method was introduced and produced promising results. As can be seen from Table 3, six algorithms were tested on eight problems, and the obtained results were recorded using six statistical indices. For each index (category variable), we have six levels or six subpopulations (algorithms) so that for each algorithm, eight values are stored, which correspond to the number of achieved scores while solving the benchmark problem. The scoring system is such that the competitive algorithms are sorted in descending order for each statistical index in terms of the best-achieved results (achieved optimum). According to the algorithms' generated rank, they are assigned scores from one to six, where score one is the worst result, while score six is the best result. For example, based on the value of the Best index in the first row of Table 2, the competitive algorithms are earned the following scores: BSGM method (6 scores), ABC (5 scores), FA (4 scores), CS (3 scores), BA (2 scores), and SA (1 score). Due to the presented results in Table 3, we can notice that the BSGM method, while solving all eight benchmark problems, has achieved 209 scores in term of the index ALL, which is indeed the highest number of scores for that index. The BSGM method for index ALL has achieved more than 56 scores compared to the rest methods. By considering the algorithms and their calculated scores shown in Table 3, it is not difficult to see that the methods SA and BA can be classified into one group due to score differences, while the rest methods such as FA, CS and BA can be put in the second group. Only the BSGM method with the most achieved scores in term of all statistical parameters (except MEAN TIME) remained isolated from the other algorithms in a separate group. Thus, based on the scores located in Table 3, we see that there are truly significant differences between the algorithms, which we will statistically confirm through hypothesis testing of equivalence of means and post hoc statistical process. In order to employ the ANOVA test, as we know, the main assumptions are population normality and homogeneity of variance. Although ANOVA is not very sensitive to normality, and the assumption of homogeneity of variance is more important for its application, we will first show with the Shapiro–Wilk test that for data in Table 3 for almost all statistical indices, a large number of subgroups (algorithms) of the dependent variable Y are normal drawn. Based on the Shapiro–Willk test for each statistical feature (BEST, MEAN, S.D., ANI*SP, MEAN TIME, ALL) and each sample of the algorithms subgroup (SA, ABC, CS, FA, BA, BSGM), we have the following estimates of statistical significance obtained by SPSS statistical software [19]:

- **BEST**: $p(ABC) > 0.05$, $p(CS) > 0.05$, and for the rest algorithms the significant p-value is less than the critical value $\alpha = 0.05$. As the significance level for the ABC and CS algorithms is greater than 0.05, the null hypothesis is retained for them while rejected for the other algorithms.
- **MEAN**: $p(FA) > 0.05$, $p(CS) > 0.05$, while for the rest methods the significance level (p-value) is less than 0.05, which implies that probably only the FA and CS algorithms retain the null hypothesis H_0.
- **S.D.**: Only for the ABC and FA algorithms we have that the significant (p-value) is greater than 0.05, which means that the algorithms SA, CS, BA and BSGM retains the alternative hypothesis H_A.

Table 3 The assignment of the scores in terms to the quality of the solutions produced by stochastic algorithms through 30 independent series

Performance measure	Metaheuristics	Engineering problems with constraints								Total scores
		P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	
BEST	SA	1	1	1	1	2	1	1	1	9
	ABC	5	5	3	6	1	3	4	3	30
	FA	4	3	5	3	5	5	5	4	34
	CS	3	4	4	5	4	4	3	6	33
	BA	2	2	2	2	3	2	2	2	17
	BSGM	6	6	6	4	6	6	6	5	45
MEAN	SA	1	2	2	2	4	1	2	2	16
	ABC	3	3	3	6	3	3	4	3	28
	FA	4	4	5	3	5	5	5	6	37
	CS	6	5	4	5	2	4	3	4	33
	BA	2	1	1	1	1	2	1	1	10
	BSGM	5	6	6	4	6	6	6	5	44
S.D.	SA	1	2	2	2	4	1	2	2	16
	ABC	3	4	3	6	2	3	4	3	28
	FA	4	3	5	3	5	5	5	6	36
	CS	5	5	4	5	3	4	3	5	34
	BA	2	1	1	1	1	2	1	1	10
	BSGM	6	6	6	4	6	6	6	4	44
ANI*SP	SA	1	1	1	1	1	1	1	1	8
	ABC	4	3	3	6	6	3	4	3	32
	FA	2	6	5	3	5	5	5	6	37
	CS	3	5	4	5	4	4	3	5	33
	BA	6	2	2	2	2	2	2	2	20
	BSGM	5	4	6	4	3	6	6	4	38
MEAN TIME	SA	3	3	3	2	2	4	4	6	27
	ABC	4	4	5	6	6	3	3	4	35
	FA	1	1	1	1	1	1	1	1	8
	CS	2	2	2	3	3	2	2	2	18
	BA	6	6	4	5	5	5	6	5	42
	BSGM	5	5	6	4	4	6	5	3	38
ALL	SA	7	9	9	8	13	8	10	12	76
	ABC	19	19	17	30	18	15	19	16	153
	FA	15	17	21	13	21	21	21	23	152
	CS	19	21	18	23	16	18	14	22	151
	BA	18	12	10	11	12	13	12	11	99
	BSGM	27	27	30	20	25	30	29	21	209

- **ANI*SP**: In this case, the significant (p-value) is greater than 0.05 only for the BSGM and CS algorithms. It implies that other algorithms reject the null hypothesis H_0.
- **MEAN TIME**: In this case, only the FA and CS algorithms do not confirm the null hypothesis H_0.
- **ALL**: Only the BA and ABC algorithms accept the alternative hypothesis H_A, so their samples are not drawn from the normal distribution.

It is important to note that for the statistical index "ALL", which best describes the differences in competitive algorithms' performance, we have that the samples of almost all algorithms (all groups or levels of the category variable) have a normal distribution. Suppose we observe the values of the dependent variable Y of the statistical index ALL at the whole group level, not partially at the level of subgroups. In that case, we can conclude based on the Shapiro–Wilk test that the significant (p-value) is drastically greater than 0.05, which indicates that the values of the dependent variable Y are normally distributed. Such normality of subgroups will not affect the ANOVA test results' accuracy since it is a very robust method, which will be demonstrated in the rest of the section. In practice, it is known that if the subgroups are equal sizes and between them, there are homogeneous variances, then the ANOVA test as a robust method is not sensitive to partial fulfilment of the assumption of normality. Therefore, based on Leven's test of homoscedasticity, let us check from data presented in Tables 3 and 4 whether the variances of the subpopulations (algorithms) are approximately equal. Due to the results of the Leven's test obtained by the statistical package SPSS, we have that:

(i) the variances are homogeneous at the statistical indices MEAN, S.D. and ALL because, for them, the significant (p-value) is greater than 0.05, so the null hypothesis is retained.
(ii) the alternative hypothesis H_A is accepted for the remaining statistical parameters.

Since we have satisfied the assumption of homogeneity for the mentioned three statistical indices, we will perform the ANOVA test for them below. Based on the data shown in Tables 3 and 4, as well as the tables for F distribution from which F critical values can be read, we get that the probability $P(F \geq F_{\alpha,k-1,N-k}|H_0\, true) = P(F \geq F_{0.05,5,42}|H_0\, true)$ belongs to the interval (0, 1E-10), so the null hypothesis H_0 can be rejected for these indices. Based on the results obtained through the SPSS software, we have the following significant values (p-value) for the mentioned statistical indices:

- **MEAN:** $SS_b = 103.750$, $SS_w = 36.250$, $MS_b = SS_b/df_b = 103.750/5 = 20.750$. Similarly, $MS_w = 0.863095$, and $F = MS_b/MS_w = 24.041379$. According to the F critical values from F distribution table, we have that probability $P(F \geq F_{\alpha,df_b,df_w}|H_0\, true) = P(F \geq 24.041379|H_0\, true)$ belongs to the interval (1E-11, 1E-10). Precisely, the statistical tool SPSS generate that p-value is equal $P(F \geq 24.041379|H_0\, true) = 2.45E-11$.
- **S.D.:** SS_b=103.0, SS_w=37.0, $MS_b = 20.60$, $MS_w = 0.880952$, $F = 23.383784$, and $P(F \geq 23.383784|H_0\, true) = 3.7281E-11$.

Table 4 The indicators of descriptive statistics for all statistical indices

Perfor. measure	Comp. methods	Mean	Std. Dev.	Std. Error	95% confidence Interval of Mean		Min	Max
					Lower bound	Upper bound		
BEST	SA	1.13	0.354	0.125	0.83	1.42	1	2
	ABC	3.75	1.581	0.559	2.43	5.07	1	6
	FA	4.25	0.886	0.313	3.51	4.99	3	5
	CS	4.13	0.991	0.350	3.30	4.95	3	6
	BA	2.13	0.354	0.125	1.83	2.42	2	3
	BSGM	5.63	0.744	0.263	5.00	6.25	4	6
MEAN	SA	2.00	0.926	0.327	1.23	2.77	1	4
	ABC	3.50	1.069	0.378	2.61	4.39	3	6
	FA	4.63	0.916	0.324	3.86	5.39	3	6
	CS	4.13	1.246	0.441	3.08	5.17	2	6
	BA	1.25	0.463	0.164	0.86	1.64	1	2
	BSGM	5.50	0.756	0.267	4.87	6.13	4	6
S.D.	SA	2.00	0.926	0.327	1.23	2.77	1	4
	ABC	3.50	1.195	0.423	2.50	4.50	2	6
	FA	4.50	1.069	0.378	3.61	5.39	3	6
	CS	4.25	0.886	0.313	3.51	4.99	3	5
	BA	1.25	0.463	0.164	0.86	1.64	1	2
	BSGM	5.50	0.926	0.327	4.73	6.27	4	6

(continued)

Table 4 (continued)

Perfor. measure	Comp. methods	Mean	Std. Dev.	Std. Error	95% confidence Interval of Mean		Min	Max
					Lower bound	Upper bound		
ANI×SP	SA	1.00	0.000	0.000	1.00	1.00	1	1
	ABC	4.00	1.309	0.463	2.91	5.09	3	6
	FA	4.63	1.408	0.498	3.45	5.80	2	6
	CS	4.13	0.835	0.295	3.43	4.82	3	5
	BA	2.50	1.414	0.500	1.32	3.68	2	6
	BSGM	4.75	1.165	0.412	3.78	5.72	3	6
MEAN TIME	SA	3.38	1.302	0.460	2.29	4.46	2	6
	ABC	4.38	1.188	0.420	3.38	5.37	3	6
	FA	1.00	0.000	0.000	1.00	1.00	1	1
	CS	2.25	0.463	0.164	1.86	2.64	2	3
	BA	5.25	0.707	0.250	4.66	5.84	4	6
	BSGM	4.75	1.035	0.366	3.88	5.62	3	6
ALL	SA	9.50	2.070	0.732	7.77	11.23	7	13
	ABC	19.13	4.643	1.641	15.24	23.01	15	30
	FA	19.00	3.546	1.254	16.04	21.96	13	23
	CS	18.88	3.044	1.076	16.33	21.42	14	23
	BA	12.38	2.446	0.865	10.33	14.42	10	18
	BSGM	26.13	3.871	1.368	22.89	29.36	20	30

- **ALL:** $SS_b = 1371.50$, $SS_w = 480.50$, $MS_b = 274.30$, $MS_w = 11.440476$, $F = 23.976275$, $P(F \geq 23.976275 | H_0\ true) = 2.553E - 11$.

The obtained p-values for statistical parameters MEAN, S.D., and ALL implies that we have significant evidence against the null hypothesis at level $\alpha = 0.05$. Therefore, the null hypothesis H_0 can be rejected, and it means that some algorithms differ statistically significantly from each other. So, it is necessary by post hoc analysis to detect where there are statistically significant differences. For instance, for statistical index ALL, we have that the SA algorithm does not differ statistically significantly from the BA algorithm, while it differs from other methods. Namely, from data shown in Tables 3 and 4, we have that Scheffe's test statistic is equal $F_S = (9.50 - 12.38)^2/(480.5/42) * (1/8 + 1/8) = 0.181251197$, and F_S is not greater than the Scheffe critical value which is given by $df_b \cdot F_\alpha(df_b, df_w) = 5 * 2.43769264 = 12.1884632$. Since for each statistical index, we have $\binom{6}{2} = 15$ algorithm comparisons, the summary of the results of Scheffe's post hoc test are shown in Table 5, where the p-values in bold indicate that there were no statistically significant differences between the pair of algorithms in their mean performance. For statistical parameter ALL, it can be observed that there are significant differences for 11 pairs of algorithms and that for our BSGM method, the null hypothesis H_0 was rejected, i.e. it indeed differs in mean performance from other algorithms. Similar inferences can be traced for the remaining two parameters, MEAN and S.D., where it can be seen in Table 5 that in eight places, there are indeed statistically

Table 5 The comparison of metaheuristics by using multicompare analysis of variance and Scheffe's post hoc test in terms of indices ALL, MEAN and S.D.

Performance measure	Algorithms	Metaheuristics				
		ABC	FA	CS	BA	BSGM
ALL	SA	0.000151	0.000188	0.000233	**0.716510**	6.0788E-10
	ABC		**1.0**	**1.0**	0.015808	0.010977
	FA			**1.0**	0.018897	0.009114
	CS				0.022528	0.007549
	BA					9.4009E-8
MEAN	SA	**0.086228**	0.000170	0.003553	**0.758651**	5.7429E-7
	ABC		**0.338422**	**0.871519**	0.001710	0.007196
	FA			**0.946488**	0.000001	**0.619528**
	CS				0.000034	**0.143798**
	BA					4.4526E-9
S.D.	SA	**0.092041**	0.000437	0.001959	**0.766473**	7.2569E-7
	ABC		**0.484974**	**0.766473**	0.001959	0.008052
	FA			**0.997777**	0.000004	**0.484974**
	CS				0.000019	**0.237264**
	BA					5.8663E-9

significant differences that indicate the rejection of the hypotheses H_0. Based on the presented statistical results, we see that the introduction of new metaheuristics such as BSGM is statistically supported. There are indeed statistically significant differences in competitive algorithms' average performance while solving a certain class of problems.

8 Conclusion and Future Work

In this paper, we used statistical multi compared tests to evaluate the performance of different meta-heuristic stochastic optimization algorithms, which provides more robust statistical results compared to the standard measures of descriptive analysis. The improved BSGM method has participated in the statistical comparison of the state-of-the-art algorithms for solving constrained engineering problems. Based on the obtained results of descriptive analysis, it has been shown that the BSGM technique is superior to the other methods. It successfully solves constrained engineering problems while preserves a low convergence rate and generates very accurate results. On the other hand, based on multiple comparative tests, it was shown that the mean performances of metaheuristic algorithms differ statistically significantly for all parameters for which they are compared. In particular, the difference between BSGM methods and other algorithms in the statistical sense came to the fore when the most robust parameter "ALL" was considered. In that case, the competing methods were clustered into three groups by the similarity of mean performance, and as a result, the BSGM method remained alone in its group. It implies that BSGM is statistically different from the other algorithms, and its introduction in solving engineering problems was cleared entirely. By examining the stated facts, it can be concluded that the BSGM method in the future can be applied for the practical solving of large-scale real-world design problems.

References

1. Alihodzic, A., Tuba, M.: Improved bat algorithm applied to multilevel image thresholding. Sci. World J. **2014**(Article ID 176718), 16 (2014). https://doi.org/10.1155/2014/176718
2. Alihodzic, A., Tuba, M.: Improved hybridized bat algorithm for global numerical optimization. In: 16th IEEE International Conference on Computer Modelling and Simulation, UKSim-AMSS 2014, pp. 57–62 (2014). https://doi.org/10.1109/UKSim.2014.97
3. Bacanin, N., Tuba, M.: Artificial bee colony (ABC) algorithm for constrained optimization improved with genetic operators. Studies Inf. Control **21**(2), 137–146 (2012). https://doi.org/10.24846/v21i2y201203
4. Bacanin, N., Tuba, M.: Firefly algorithm for cardinality constrained mean-variance portfolio optimization problem with entropy diversity constraint. Sci. World J. **2014**, 115–139 (2014). https://doi.org/10.1155/2014/721521

5. Brajevic, I., Tuba, M.: An upgraded artificial bee colony algorithm (abc) for constrained optimization problems. J. Intell. Manuf. **24**(4), 729–740 (2013). https://doi.org/10.1007/s10845-011-0621-6
6. Brown, M.B., Forsythe, A.B.: Robust tests for the equality of variances. J. Amer. Stat. Assoc. **69**(346), 364–367 (1974). https://doi.org/10.1080/01621459.1974.10482955
7. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006). http://jmlr.org/papers/v7/demsar06a.html
8. Derrac, J., García, S., Hui, S., Suganthan, P.N., Herrera, F.: Analyzing convergence performance of evolutionary algorithms: a statistical approach. Inf. Sci. **289**, 41–58 (2014). https://doi.org/10.1016/j.ins.2014.06.009
9. Eftimov, T., Korošec, P.: Identifying practical significance through statistical comparison of meta-heuristic stochastic optimization algorithms. Appl. Soft Comput, **85**, 105, 862 (2019). https://doi.org/10.1016/j.asoc.2019.105862
10. Eftimov, T., Korošec, P.: A novel statistical approach for comparing meta-heuristic stochastic optimization algorithms according to the distribution of solutions in the search space. Inf. Sci. **489**, 255–273 (2019). https://doi.org/10.1016/j.ins.2019.03.049. https://www.sciencedirect.com/science/article/pii/S0020025519302610
11. Fister, I., Fister, J., Yang, X., Brest, J.: A comprehensive review of firefly algorithms. Swarm Evol. Comput. **13**(1), 34–46 (2013). https://doi.org/10.1016/j.swevo.2013.06.001
12. Gandomi, A.H., Yang, Alavi, A.H., Talatahari, S.: Bat algorithm for constrained optimization tasks. Neural Comput. Appl. **22**(6), 1239–1255 (2013). https://doi.org/10.1007/s00521-012-1028-9
13. Gandomi, A.H., Yang, X.S., Alavi, A.H.: Mixed variable structural optimization using Firefly Algorithm. Comput. Struct. **89**(23–24), 2325–2336 (2011). https://doi.org/10.1016/j.compstruc.2011.08.002
14. Gandomi, A.H., Yang, X.S., Alavi, A.H.: Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng. Comput. **29**(1), 17–35 (2013). https://doi.org/10.1007/s00366-011-0241-y
15. García, S., Herrera, F.: An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J. Mach. Learn. Res. **9**(89), 2677–2694 (2008). http://jmlr.org/papers/v9/garcia08a.html
16. García, S., Molina, D., Lozano, M., Herrera, F.: A study on the use of non-parametric tests for analyzing the evolutionary algorithms? behaviour: a case study on the cec2005 special session on real parameter optimization. J. Heuristics **15**(6), 617–644 (2008). https://doi.org/10.1007/s10732-008-9080-4
17. shi He, X., Ding, W.J., Yang, X.S.: Bat algorithm based on simulated annealing and Gaussian perturbations. Neural Comput. Appl. **25**(2), 459–468 (2013). https://doi.org/10.1007/s00521-013-1518-4
18. Hogg, R.V., McKean, J.W., Craig, A.T.: Introduction to Mathematical Statistics, 8th edn. Pearson, London (2019)
19. IBM Corp.: IBM SPSS Statistics for Windows. https://hadoop.apache.org
20. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report - TR06, pp. 1–10 (2005)
21. Keselman, H., Rogan, J.: A comparison of the modified-tukey and scheffe methods of multiple comparison for pairwise contrasts. J. Amer. Stat. Ass. - J AMER STATIST ASSN **73**, 47–52 (1978). https://doi.org/10.1080/01621459.1978.10479996
22. Kirkpatrick, S., Jr., C.G., Vecchi, M.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983). https://doi.org/10.1126/science.220.4598.671
23. Long, W., Liang, X., Huang, Y., Chen, Y.: An effective hybrid cuckoo search algorithm for constrained global optimization. Neural Comput. Appl. **25**(3–4), 911–926 (2014). https://doi.org/10.1007/s00521-014-1577-1
24. Nigdeli, S.M., Bekda, G., Yang, X.S.: Application of the flower pollination algorithm in structural engineering. Model. Optim. Sci. Technol. **7**, 25–42 (2015). https://doi.org/10.1007/978-3-319-26245-1_2

25. Shapiro, S.S., Wilk., M.B.: An analysis of variance test for normality (complete samples). Biometrika **52**(3/4), 591–611 (1965). https://doi.org/10.2307/2333709
26. Shilane, D., Martikainen, J., Dudoit, S., Ovaska, S.J.: A general framework for statistical performance comparison of evolutionary computation algorithms. Inf. Sci. **178**(14), 2870–2879 (2008). https://doi.org/10.1016/j.ins.2008.03.007
27. Tuba, M., Alihodzic, A., Bacanin, N.: Cuckoo search and bat algorithm applied to training feed-forward. Neural Netw. **585**, 139–162 (2014). https://doi.org/10.1007/978-3-319-13826-8_8
28. Tuba, M., Bacanin, N., Alihodzic, A.: Firefly algorithm for multi-objective RFID network planning problem. Telecommun. Forum Telfor (TELFOR) 95–98 (2014). https://doi.org/10.1109/TELFOR.2014.7034365
29. Tuba, M., Jovanovic, R.: Improved ant colony optimization algorithm with pheromone correction strategy for the traveling salesman problem. Int. J. Comput. Commun. Control **8**(3), 477–485 (2013). https://doi.org/10.15837/ijccc.2013.3.7
30. Črepinšek, M., Liu, S.H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput. Surv. **45**(3), 35:1–35:33 (2013). https://doi.org/10.1145/2480741.2480752
31. Yang, X.S.: A new metaheurisitic bat-inspired algorithm. Studies Comput. Intell. **284**, 65–74 (2010). https://doi.org/10.1007/978-3-642-12538-6_6
32. Yang, X.S.: Review of meta-heuristics and generalised evolutionary walk algorithm. Int. J. Bio-Inspired Comput. **3**(2), 77–84 (2011). https://doi.org/10.1504/IJBIC.2011.039907
33. Yang, X.S.: Efficiency analysis of swarm intelligence and randomization techniques. J. Comput. Theor. Nanosci. **9**(2), 189–198 (2012). https://doi.org/10.1166/jctn.2012.2012
34. Yang, X.S.: Free lunch or no free lunch: That is not just a question? Int. J. Artif. Intell. Tools **21**(3), 5360–5366 (2012). https://doi.org/10.1142/S0218213012400106

Heuristic Approaches for the Stochastic Multi-depot Vehicle Routing Problem with Pickup and Delivery

Brenner H. O. Rios, Eduardo C. Xavier, Flávio K. Miyazawa, and Pedro Amorim

Abstract The stochastic multi-depot vehicle routing problem with pickup and delivery (S-MDVRPPD) is a new problem presentend in this work. The deterministic variation of S-MDVRPPD constitutes a generalization of the Traveling Salesman. In the S-MDVRPPD the pickups and delivery points are present in a route with some probability. A route for each depot must be satisfy the following restrictions: (1) each cycle starts and ends at the corresponding vehicle's depot; (2) each node is visited exactly once by one vehicle; (3) each pair of pickup and delivery points must belong to the same cycle; and (4) each pickup point in a cycle appears before its delivery pair. The objective is to find a solution with the minimum expected cost. We use a closed-form expression to compute the expected cost of an a priori route under general probabilistic assumptions. A linear integer programming model is presented for the deterministic version of S-MDVRPPD. To solve the S-MDVRPPD we propose a Variable Neighborhood Search (VNS) and Iterated local search (ILS) that use the Variable Neighborhood Descent (VND) as local search procedure. We compare the performance of the proposed algorithms with a Tabu Search algorithm (TS). We evaluate the performance of these heuristics on a data set adapted from TSPLIB instances. The results show that the VNS proposed is efficient and effective to solve S-MDVRPPD.

Keywords Heuristic · Multi-depot vehicle routing · Integer programming

B. H. O. Rios (✉) · E. C. Xavier · F. K. Miyazawa
Institute of Computing, University of Campinas, Campinas, Brazil
e-mail: brenner@students.ic.unicamp.br

E. C. Xavier
e-mail: ecx@ic.unicamp.br

F. K. Miyazawa
e-mail: fkm@ic.unicamp.br

P. Amorim
Faculty of Engineering, University of Porto, Porto, Portugal
e-mail: amorim.pedro@fe.up.pt

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_2

1 Introduction

The vehicle routing problem (VRP) consists of designing a set of routes optimally, such that each route begins and ends in a central depot, and where a set of geographically dispersed customers are visited. The VRP is subject to several constraints, such as fleet size, vehicle capacity, time windows, precedence relations between customers, etc. The economic importance of this problem is perceived daily in current business models and new concepts of logistics operations around the world. A good example is the segment of business-to-consumer crowdsourced services, such as online food ordering services and peer-to-peer ride-sharing. According to Morgan Stanley Research (2020), the online food delivery market could grow to $470 billion in 2025. In relation to peer-to-peer ride-sharing, according to McKinsey (2017), Uber and Lyft presented a revenue of $10 billion in 2016 only in the US.

A well-known generalization of the VRP is the Multi-Depot Vehicle Routing Problem (MDVRP) [6]. In the standard version of MDVRP, each customer is visited by a vehicle that begins and ends at the same depot. On the other hand, the VRP with pickup and delivery (VRPPD) is a problem where the service required by customers can be pickup and delivery of commodities [15]. In this work, we study the stochastic version of MDVRP with pickups and deliveries (MDVRPPD). The principle of MDVRPPD is to design a collection of optimal routes for a fleet of vehicles, where each vehicle is located in a single depot. In the MDVRPPD, the routes allow serving geographically dispersed customers that can be delivery or pickup points. The vehicle fleet size is equal to the number of depots, which means that there is only one vehicle in each depot. Consequently, each route begins and ends at the same depot. Finally, the vehicles must visit each customer only once. Figure 1 shows the routes of three vehicles, each one belonging to a single depot.

The stochastic VRP (SVRP) is any VRP where one or more parameters are stochastic. In the stochastic version of the MDVRPPD, the pickup and delivery points are uncertain. Consider the situation where third-parties sell products from

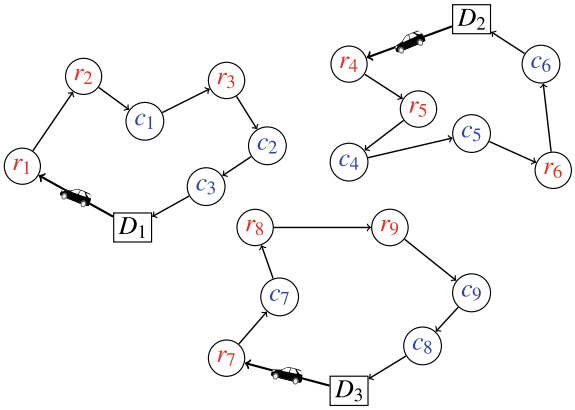


Fig. 1 Example of a solution for the MDVRPPD with three vehicles. Each vertex r_i represents a pickup point while c_i represents its corresponding delivery point

an online marketplace provider (e-commerce platform owned and operated by the provider). Usually, the provider is responsible for collecting and delivering the products sold (for example, the food delivery service provided by UberEats). Based on previous data, it is possible to create better routes since the provider has access to the probability distribution of a request from customer A to seller B.

We propose an Variable Neighborhood Search (VNS) and Iterated Local Search to solve this problem that use the Variable Neighborhood Descent (VND) heuristic as a local search. We denote the proposed VNS algorithm by GVNS (Generalized VNS). The GNVS is based on the VNS heuristics presented by [5] for the Pickup and Delivery Traveling Salesman Problem with LIFO Loading. On the other hand, we denote the proposed ILS by ILS-VND. To evaluate the performance of GVNS and ILS-VND, we compare them with an adaptation of the TABUSTOCH algorithm proposed by Gendreau et al. [11]. The tabu search, VNS and ILS are some of the main methods to deal with SVRPs [27].

The remainder of this paper is organized as follows. Section 2 presents the description and mathematical formulation of the MDVRPPD. Section 3 presents the closed-form expression to compute the expected cost of an a priori route. Section 4 presents the basic operators used by the VNS and ILS algorithms. Section 5 introduces the TS heuristic proposed for the problem. Section 6 describes the computational experiments, and Sect. 7 presents the conclusions of this work.

1.1 Related Work

The proposed problem has a close connection with the well known multi-depot traveling salesman problem (mTSP) [16]. Specifically, with the special case of multi-depot multiple travelling salesman problem (MmTSP). In the MmTSP each salesman starts from a unique city, travels to a set of cities and completes the route by returning to his original city with each city visited once [2]. Kara and Bektas [3] presents a mTSP review and explores connections with VRPs. The MDVRPPD is also closely related to the steiner multi cycle problem (SMCP), recently introduced in [24]. The SMCP arises in the scenario where a company has to periodically exchange goods between two different locations, and different companies can collaborate to create a route that visits all its pairs of locations sharing the total cost of the route [24]. The MDVRPPD can be seen as a version of the SMCP with depots.

There are several heuristic approaches to solving VRPs and its stochastic variant. State of the art solutions include: particle swarm optimization approach [7], VNS [25], adaptive large neighbourhood search [18], ant colony optimization [31], genetic approach [20], tabu search [9], ILS [28], simulated annealing [12] and hybrid heuristic with exact methods [29, 30]. A review of the solution methods used in the past 20 years for the SVRP is presented in [23].

2 Problem Description and Model

In this section, we present the MDVRPPD and model it as an integer linear program first, then we define the S-MDVRPPD.

2.1 MDVRPPD

In this work, MDVRPPD is defined as follows. Let $G = (V, E)$ be a complete undirected graph, where $V = \{v_1, \ldots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) : v_i, v_j \in V, i < j\}$ is the edge set. With each edge (v_i, v_j), it is associated a non-negative cost or distance d_{ij}. A subset of vertices $D = \{v_1, \ldots, v_m\}$ represents the depots, and the remaining vertices $V' = \{v_{m+1}, \ldots, v_n\}$ corresponds to pickup and delivery points. Let $w = |V'|/2$, then w vertices are pickup points and w vertices are delivery points. Each pickup point v_i is associated with a unique delivery point v_{i+w}, and vice versa, for $m + 1 \leq i \leq m + w$. There are m identical vehicles of unlimited capacity such that each one is located in a single depot. Each vehicle leaves its depot, serves a subset of pickup and delivery vertices and returns to its depot, forming a cycle (or route). The problem consists in determining a set of m vehicle cycles of minimal total cost considering the following constraints: (a) each cycle starts and ends at the corresponding vehicles depot; (b) each $v \in V'$ is visited exactly once by one vehicle (c) each pair of pickup and delivery points, e.g $\{v_i, v_{i+w}\}$ for $m + 1 \leq i \leq m + w$, must belong to the same cycle and (d) each cycle has an orientation where each pickup vertex in this cycle appears before its delivery pair.

The MDVRPPD is NP-hard since it includes the Traveling Salesman Problem (TSP) as a special case (e.g. if each pair of pickup and delivery are in the same location and there is only one depot).

We adapt the mathematical formulation proposed in [16] for the deterministic static version of MDVRPPD. In this formulation we assume G is a complete symmetric directed graph. The parameters and variables of the formulation are defined in Table 1.

Table 1 Parameters and decision variables for the MDVRPPD

D	Set of depots, $\{v_1, \ldots, v_m\}$
V'	Set of nodes (pickup and delivery), $\{v_{m+1}, \ldots, v_n\}$
H^+	Set of pickup nodes, $\lvert H^+\rvert = w$
u_i^k	A positive integer variable that indicates the order vertex i is visited by vehicle k, and $u_i^k = 0$ if i is not visited by k, $i \in V'$, $k \in D$
d_{ij}	Distance between vertices i and j
x_{ij}^k	If the vehicle from depot k travel along arc (i, j), then $x_{ij}^k = 1$, otherwise $x_{ij}^k = 0$

$$\text{minimize} \sum_{k \in D} \sum_{j \in V'} (d_{kj} x_{kj}^k + d_{jk} x_{jk}^k) + \sum_{k \in D} \sum_{i \in V'} \sum_{j \in V'} d_{ij} x_{ij}^k \tag{1}$$

$$s.t. \quad \sum_{j \in V'} x_{kj}^k = 1, \quad k \in D, \tag{2}$$

$$\sum_{j \in V'} x_{jk}^k = 1, \quad k \in D, \tag{3}$$

$$\sum_{k \in D} x_{kj}^k + \sum_{k \in D} \sum_{i \in V'} x_{ij}^k = 1, \quad \forall j \in V', \tag{4}$$

$$x_{kj}^k + \sum_{i \in V'} x_{ij}^k = x_{jk}^k + \sum_{i \in V'} x_{ji}^k, \quad \forall k \in D, j \in V', \tag{5}$$

$$u_i^k \leq n \left(\sum_{j \in V'} x_{ij}^k + x_{ik}^k \right), \quad i \in V', \quad k \in D, \tag{6}$$

$$x_{ki}^k \leq u_i^k, \quad i \in V', \quad k \in D, \tag{7}$$

$$u_i^k + 1 \leq u_j^k + (1 - x_{ij}^k) n, \quad i, j \in V', \quad k \in D, \tag{8}$$

$$u_i^k + 1 \leq u_{i+w}^k + \left(1 - \sum_{j \in V'} x_{ij}^k + x_{ik}^k \right) n, \quad i \in H^+, \quad k \in D, \tag{9}$$

$$x_{ij}^k \in \{0, 1\}, \quad i, j \in V, \tag{10}$$

$$u_i^k \in \mathcal{Z}^+, \quad i \in V, \quad k \in D, \tag{11}$$

In this formulation, constraint (2) ensures that exactly one vehicle depart from each depot $k \in D$, while (3) assures the vehicle returns to the depot. Constraint (4) ensures that each node is visited exactly once. Route continuity is ensured by the flow conservation constraints (5). Constraints (6) assures that the order of client i is 0 if it is not in the route of vehicle k. Constraints (7) impose that if i is the first vertex visited in route k, than its order in this route is at least 1. Constraint (8) is a subtour elimination constraint, since if j is visited after i in route k, then the visit order of j must be larger than the one of i in this route. Constraint (9), ensures that each pick-up

node (i) must be visited before the corresponding delivery node $(i + w)$. Finally we have the integrality constraints (10) and (11) of the variables in the model.

2.2 S-MDVRPPD

Now, we define the particular S-MDVRPPD considered in this work. This problem has one type of uncertainty: stochastic pickup and delivery points. Each pair $\{v_i, v_{i+w}\} \in V'$, for $m + 1 \leq i \leq m + w$, has a probability p_i of being present when traveling along the route. When pickup point v_i is absent, delivery point v_{i+w} is also absent. We consider the S-MDVRPPD as a two stage stochastic problem. In the *first stage*, a set of cycles satisfying constraints (a)–(d) of the MDVRPPD are computed. The presence or absence of $\{v_i, v_{i+w}\}$ is revealed at the latest time upon leaving the preceding vertex of v_i. We suppose that the demand of every delivery point v_{i+w} is the same e.g. one unit. In the *second stage*, the first stage routes are followed as planned, with the following exception: any absent node is skipped. The S-MDVRPPD consists of designing a first stage solution that *minimizes the expected cost of the second stage solution*.

The S-MDVRPPD can be formulated as a stochastic integer program. We will use the parameters and variables defined in Table 1. Let $T(x, \xi)$ be the cost of second stage solution if $x = (x_{ij}^k)$ is the first stage solution, and $\xi = (\xi_i)$ is the vector of non-negative random variables associated with the vertices of V'. The S-MDVRPPD is then formulated as

$$\min_x E_\xi[T(x, \xi)] \tag{12}$$

subject to Eqs. (2)–(9).

3 The Expected Cost of an a Priori Route

Given a priori computed route $s = (v_0, v_1, \ldots, v_{2q}, v_0)$, where v_0 is a depot, let l_s be the cost/length of s. Our goal is to compute efficiently the expected length $E[l_s]$ of route s, given that during its execution, each pair $\{v_i, v_{i+w}\}$ of pickup and delivery points in this route have a probability of occurring during s's execution. We may also refer to node v_i as r_i, and v_{i+w} as c_i. Let $P(v_i)$ be the probability that node v_i appears in s. Note that we have the following relationship for a pair of pickup and delivery points r_i and c_i: $P(r_i) = P(c_i)$, $P(c_i|r_i \text{ appears}) = P(r_i|c_i \text{ appears}) = 1$, and $P(c_i|r_i \text{ not appear}) = P(r_i|c_i \text{ not appear}) = 0$.

In this theorem we assume that R is the set of pickup points that appear in s $(|R| = q)$. The pickup points are numbered in the superscript, in the order they appear in s, from r^1 until r^q. Likewise, C is the set of the corresponding delivery points and are also numbered in the superscript, from c^1 until c^q in the order they

appear in s. We also use the following notation. If v_i is a pickup point we denote this by writing $r(v_i)$, and its corresponding delivery point as v_i^-, and if v_i is a delivery node, we denote this by writing $c(v_i)$ and denote its corresponding pickup point as v_i^+. Finally, given a subsequence $s_i^j = (v_i, v_{i+1}, \ldots, v_j)$ of s, let $R(s_i^j)$ denote the set containing the pickup points that appear in s_i^j, and also containing the pickup points of the delivery vertices that appear in s_i^j. Notice that $|R(s_i^j)| \leq j - i + 1$, and it is strictly small only when a pickup point appears in s_i^j and its delivery point also appears. Then we can compute $E[l_s]$ as follows.

Theorem Given a priori route $s = (v_0, v_1, \ldots, v_{2q}, v_0)$, then:

$$\begin{aligned} E[l_s] = &\sum_{i=1}^{q} d_{v_0,r^i} P(r^i) \prod_{k=1}^{i-1} (1 - P(r^k)) \\ &+ \sum_{i=1}^{q} d_{c^i,v_0} P(c^i) \prod_{k=i+1}^{q} (1 - P(r^k)) \\ &+ \sum_{i=1}^{2q} \sum_{j=i+1}^{2q} f(v_i, v_j) \end{aligned} \tag{13}$$

where

$$f(v_i, v_j) = \begin{cases} 0 & \text{, (a) or (b)} \\ d_{v_i,v_j} P(v_i) \prod_{v \in R(s_{i+1}^{j-1})} (1 - P(v)) & \text{, (c)} \\ d_{v_i,v_j} P(v_i) P(v_j) \prod_{v \in R(s_{i+1}^{j-1})} (1 - P(v)) & \text{, (d)} \end{cases} \tag{14}$$

Proof In Eq. (13) we are basically computing the probability of each edge between vertices in s to appear, in the execution of s, and multiplying this probability by the edge's cost. We have three terms in this equation.

In the first term, we are computing the expected cost of each possible initial edge of the route, such that if the route starts with (v_0, r^i) then all previous pickup points r^k, $k = 1, \ldots, i - 1$ must not be present in the route. In the second term, we are computing the expected cost of each possible final edge in the route, similar to the first term.

In the last term we compute the expected cost of each edge from any pair of vertices in the route. Figure 2 represents all the possible cases between vertices v_i and v_j. Since cases (a) or (b) do not occur in practice the expected cost of an edge (v_i, v_j) in any one of these cases is zero. In case (c), the probability of going directly from $r(v_i)$ to $v_j = v_i^-$ is the probability of request of pickup point v_i to occur times the probability of requests of points in between v_i and v_j to not occur. Likewise, in case (d), if v_i and v_j are not related in any of the previous cases, then the probability of edge (v_i, v_j) to occur, is equal to the probability of v_i and v_j to occur times the probability of none of the requests of vertices in between them to occur. □

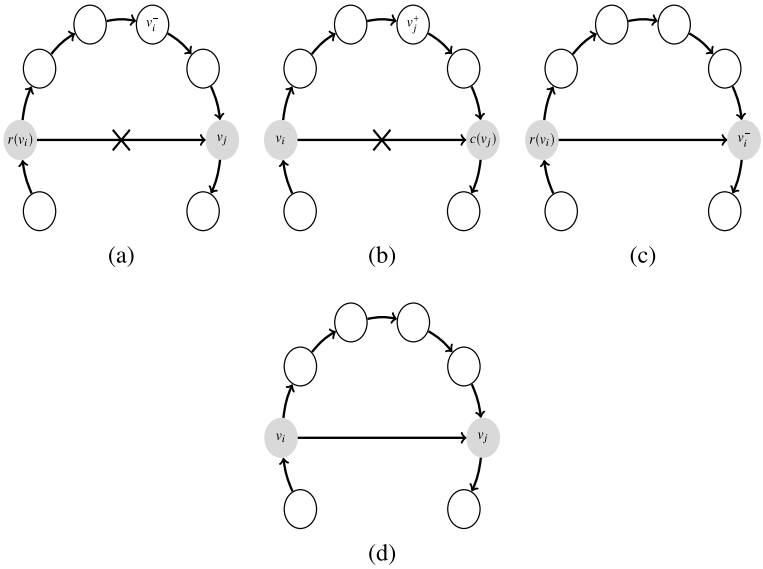


Fig. 2 In **a** v_i is a pickup node and v_j appears after v_i's corresponding delivery node v_i^-. In **b** v_i is any vertex, v_j is a delivery node and v_j^+ appears after v_i. Both situations, **a** and **b** do not occur, since in **a**, if $r(v_i)$ is present in the route then v_i^- must appear as well, and in **b** if $c(v_j)$ is present then v_j^+ must be present as well, so going from v_i directly to v_j skipping vertices in between is not a valid route. In **c** we have the case where v_i is a pickup point and v_j is its corresponding delivery point. In **d** we have all other cases that do not belong to one of the previous cases

We can compute the expected cost of an a priori route, $E(l_s)$, with time complexity $O(q^2)$, where $2q + 2$ is the size of the route.

4 ILS and VNS

In this section, we introduce the local search operators, perturbation operators, and heuristics used by VNS and ILS. Then we present both algorithms proposed in this work to deal with the S-MDVRPPD.

4.1 Initial Solution Generation

The method employed for building a feasible initial solution is based on the work of [17], so it is generated by following two steps. The first step is called *customer assignment*. In this step, each associated pickup and delivery is assigned to a depot. Then each pickup and delivery pair is visited by a vehicle that starts from that

specific depot. The set of vertices formed by a depot and its associated pairs of pickup and delivery is called a group. After the groups' construction, the second step called *customer sequencing* is carried out, then the service sequence of the pickup and delivery points in each group is decided. By following the two steps above, all vehicles' routes can be constructed and form the initial solution for S-MDVRPPD. The details of these two steps are introduced in the following paragraphs.

4.1.1 Nodes Assignment

This step assigns pickup and delivery points associated with a depot conveniently close to them. This procedure adapts the method proposed by [17], which assigns clients to depots based on probabilities. The idea is to make the initial solution more flexible. Let $d(D_a, r_i, c_i)$ be the sum of the distance from r_i to D_a and the distance from c_i to D_a, and $\overline{d(D, ri, ci)}$ be the average distance of a pair of pickup and delivery associated $\{r_i, c_i\}$ to all depots. The probability of a pickup and delivery pair being assigned to a depot is calculated by Eq. (15).

$$P(D_a, r_i, c_i) = \frac{\max\left\{\overline{d(D, r_i, c_i)} - d(D_a, r_i, c_i), 0\right\}}{\sum_{a=1}^{|D|} \max\left\{\overline{d(D, r_i, c_i)} - d(D_a, r_i, c_i), 0\right\}} \tag{15}$$

From Eq. (15), a pickup and delivery pair can be assigned to a depot that is not necessarily the closest to them. For example, in Fig. 3 the distances between $\{r_1, c_1\}$, and depots 1, 2 and 3 are 3, 5, 9, respectively. We denote an associated pickup and delivery points by $\{r_1, c_1\}$. The average distance between the pickup and delivery pair is 2.666. The probabilities of the pickup and delivery pair being assigned to deposits 1, 2 and 3 are 81.0%, 20.83% and 0.00%, respectively. Note that depot 1 is the closest to the pickup and delivery pair; however, there is a probability that the pair of vertices is assigned to deposit D_2.

4.1.2 Nodes Sequencing

This step decides the service sequence of the pickup and delivery pairs in a group. The goal is to sequence the nodes to create cycles. Suppose an instance with a single depot. Let D be a depot, and t a tour that initially contains D. The method computes the distance from each node to the depot D. Suppose that v_1 is the node closest to D, then v_1 is selected and added to t $(D - v_1)$. Then the distance of v_1 to each node that does not belong to the tour t is computed. Suppose v_2 is the node closest to v_1, then v_2 is added to t after v_1 $(D - v_1 - v_2)$. The process continues until all nodes are added to t. Figure 4 shows an illustrative example of group sequencing. The complexity of this procedure is $O(n^2)$.

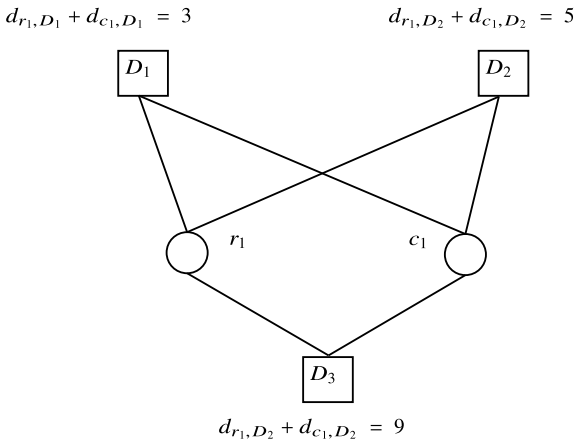


Fig. 3 Example of nodes assignment

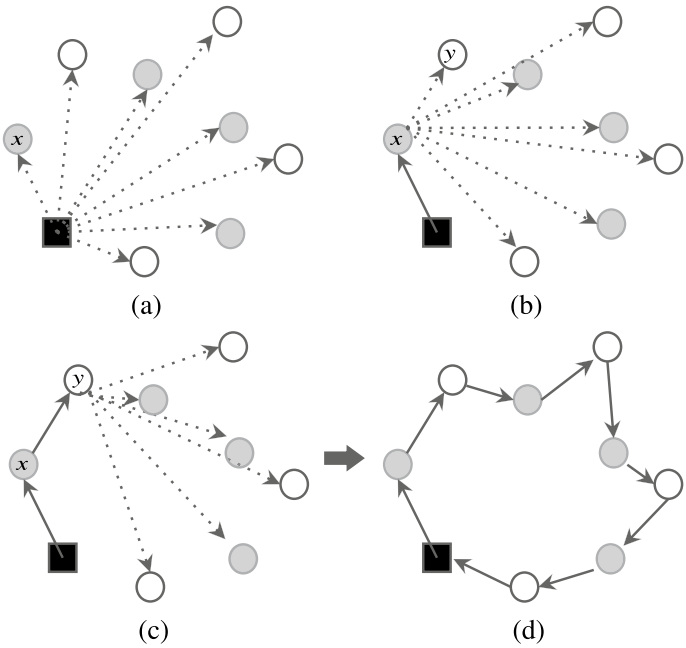


Fig. 4 Example of nodes sequencing in a route. Gray nodes are pickups and white nodes deliveries. In **a** distances from the depot to all vertices are calculated. In **b** node x is added to the route, since it is the closest to the depot and its addition does not break the constraints of the problem. Then, the distances from x to all the nodes that are not part of the route are computed. The node y is the closest to x, and its addition does not break the constraints of the problem. In **c** node y is added to the route. The process repeats until all nodes are added to the route. The obtained route is shown in (**d**)

4.2 Local Search

The local search procedure that aims to improve the quality of the initial solution is based on the VND algorithm. Mladenović and Hansen [21] proposed the VND as a procedure that systematically modifies the neighborhood structure in a deterministic way.

The proposed procedure considers six neighborhood structures $\{N_1, \ldots, N_6\}$ used to perform movements between pairs of pickups and deliveries that belong to the same route or different routes. Only valid movements are allowed, e.g., movements that do not violate the precedence constraint (a pickup node always appears before that the delivery associated with it). These operators are exhaustively executed. We divide the operators into three groups: inter-cycle (movements of nodes between different cycles), intra-cycle (movements of nodes in the same cycle), and inter&intra-cycle (movements where the cycle to which the nodes belong is not relevant). The inter-cycle operators are shift(1, 0) and swap(1, 1). The intra-cycle operators are 2-opt, 3-opt, and reverse. Finally, Mix-shift(1, 0) is the only operator that is inter&intra-cycle. In the inter&intra-cycle operators, each removed node can only be inserted before or after its p closest neighbors (these neighbors can belong to the same cycle or to different cycles). The list of neighborhoods considered are:

Shift(1, 0)—$N^{(1)}$—A vertex v (e.g., pickup i^+) is removed from a route t_1 to be inserted into route t_2, then the pair vertex of v (e.g., delivery i^-) is moved to the best position in route t_2 that does not break the constraints of the problem. This operator runs in $O(n)$. Figure 5 presents an example of this operator.

Swap(1, 1)—$N^{(2)}$—Exchange between a vertex v (e.g., pickup i^+) from a route t_1 and vertex u (e.g., pickup j^+) from a route t_2. The pair of vertex of v (e.g., delivery i^-) is moved to the best position of route t_2 that does not break the problem's constraints.

Algorithm 1: VND

```
Let r be the number of neighborhoods structures and s a
current solution;
k := 1; current neighborhood;
while k ≤ r do
    Find the best neighbor s' of s ∈ N^k ;
    if f(s') < f(s) then
        s := s' ;
        k := 1 ;
        intensification in the modified routes;
        s' := 2 − opt(s) ;
        s'' := 3 − opt(s') ;
        s''' := Reverse(s''') ;
        if f(s''' ≤ f(s) then
            s := s'''
    else k := k + 1 ;
```

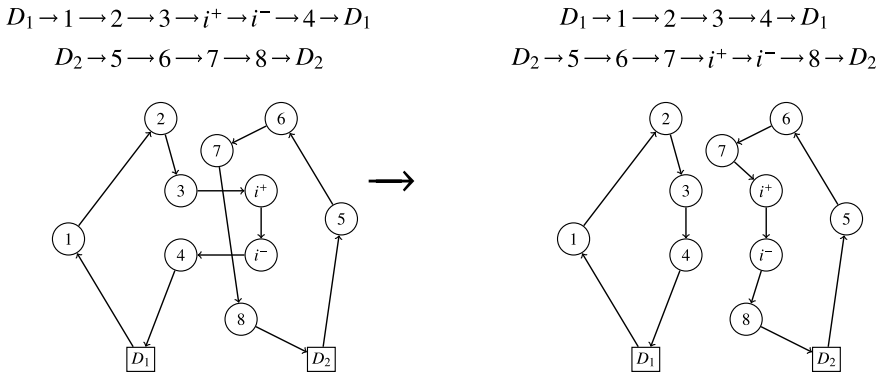


Fig. 5 An example of nodes move operator

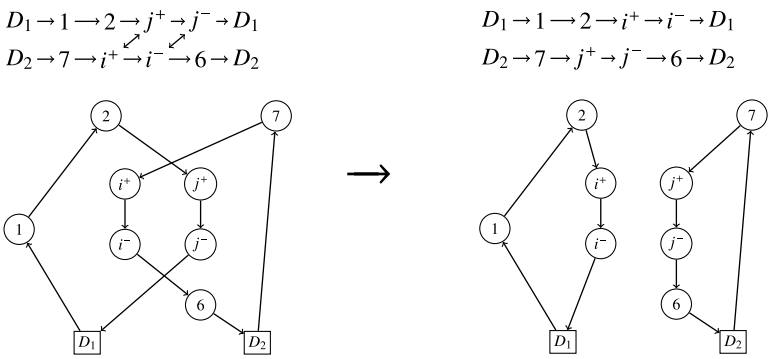


Fig. 6 An example of nodes exchange operator

Analogously, the pair of vertex u (e.g., delivery j^-) is moved. This operator runs in $O(n)$. Figure 6 presents an example of this operator.

In case of improvement of the current solution, we will initiate an intensification process on each route. The objective is to decrease the cost of each route to reduce the value of the current solution. Therefore, the following neighborhoods are explored.

2-opt—$N^{(3)}$—Two nonadjacent arcs are removed and another two are added to form a new route. We only consider movements that do not break the constraints of the problem. This operator runs in $O(n^3)$, see [5].

3-opt—$N^{(4)}$—Three nonadjacent arcs are removed and another three are added to form a new route. We only consider movements that do not break the constraints of the problem. This operator runs in $O(n^4)$, see [5].

Reverse—$N^{(5)}$—This operator reverses the direction of the route. Then swaps are performed between each pair of pickup and delivery. This operator runs in $O(n)$ (Fig. 7).

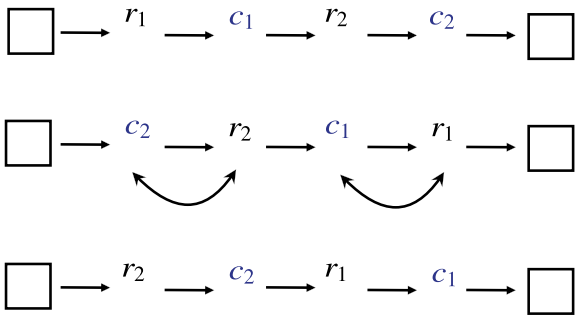


Fig. 7 Example of reverse operator

Mix-Shift(1, 0)—$N^{(6)}$—This operator is similar to the Shift(1, 0) operator with the difference that here is allowed movement within its own route. This operator runs in $O(n)$.

Based on the neighborhood structure proposed by Gendreau et al. [11] and on the Mix-shift operator(1, 0), in this work the Random Mix-Shift heuristic is proposed (Algorithm 2). This heuristic selects an associated pickup and delivery pair $\{r, c\}$ at random, then the pickup point r is removed and inserted immediately before or after one of its p closest neighbors. Then the delivery point c is removed and inserted randomly in the same route respecting the restrictions of the problem. To avoid necessary iterations of the Random Mix-Shift heuristic, we chose to use s' if it is *promising*. A solution s' is *promising* if it has the potential to become the new best solution, specifically, if its cost is at most $\alpha\%$ higher than the cost of the best solution so far, s. The idea behind using s' is to avoid quickly reaching the number of iterations necessary to finish the execution of the heuristic. We use this heuristic in the GVNS and ILS-VND as a refinement mechanism to improve the initial solution.

Algorithm 2: Random Mix-Shift

```
for k := 1 ..., MaxIterShift do
    r, c := SelectRandomPair(r, c) ;
    s' := Mix-Shift(s, r, c) ;
    if f(s') < f(s*) then
        s := s'; s* := s'; k := 1 ;
    else
        if f(s') < αf(s*) then
            s := s' ;
        else
            s := s*;
```

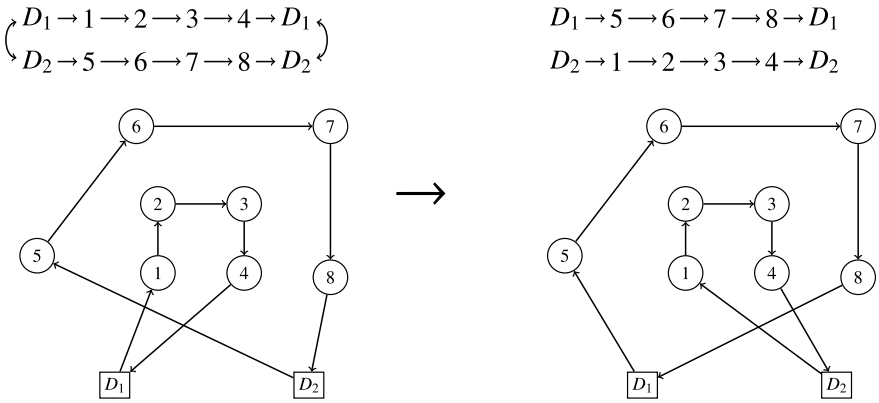


Fig. 8 An example of depots exchange operator

4.3 Perturbation Operators

In this subsection, we introduce a set $\{P^{(1)}, P^{(2)}\}$ of two operators used to perturb the solutions. Other perturbation mechanisms that could be adapted for S-MDVRPPD can be found in [5]. It is important to note that other perturbation operators were analyzed, such as double bridge [5]. Such operators were not successful, in many cases these operators perform movements that generate unfeasible solutions in addition to being computationally expensive. Every time the *Shake* method is called, one of the following perturbation operators is randomly selected.

Double-Swap—$P^{(1)}$—Two Swap(1, 1) operators are performed in sequence randomly. This operator runs in $O(n)$.

Depot Exchange—$P^{(2)}$—The depot exchange operator selects two depots at random, then exchange their positions. This operator runs in $O(1)$. Figure 8 presents an example of this operator.

4.4 VNS

Variable neighborhood search (VNS) was proposed by Mladenovic and Hansen [21]. The idea is to apply a perturbation to the search operator when a local minimum is reached. This perturbation reactivates the search to reach a solution that could not have been found with the current search operator, also allows further exploration of the solution space. VNS generally consists of three main steps: shaking step, local search step, and neighborhood change step [14]. The local search step aims at finding a local optimal solution in a neighborhood. The shaking step aims at escaping from a local optimal solution as well as generating a new initial solution for the local search step [26]. Several enhancements of the original method have since been proposed

in [13]. The VNS is summarized in Algorithm 3, where s^* corresponds to the best solution and $k_{\max}$ is the maximum number of neighborhoods.

Algorithm 4 presents the pseudo-code of the GVNS for S-MDVRPPD. The algorithm begins by initializing the parameters: $diversification$ and $MaxIter$. An initial solution s is generated by the $FindInitialSolution$ heuristic. The Random Mix-Shift heuristic is used to refine the initial solution. The main loop iterates until the maximum number of executions $MaxIter$ is reached. A perturbation procedure called $Shake$ is applied to the current solution s_i, generating a solution s_i'. If the solution s_i' has a lower cost than the best solution s^*, then s^* is updated with s_i' and the iteration counter $iter$ becomes 0. The local search (VND) performed on the solution s_i' generates a solution s_k. If the solution s_k has a lower cost than the best solution s^*, then we update s^* with the solution s_k and the iteration counter becomes 0. If the local search is not able to improve the best solution, we use method $AcceptanceCriteria$ to accept s_k as the new current solution. Finally, the diversification phase is carried out; this phase is performed only once during the algorithm's execution. In the diversification phase the iteration counter $iter$ is set to 0, and a perturbation to the current solution is performed.

Algorithm 3: Gereralized VNS

```
s ← FindInitialSolution() ;
s1 ← LocalSearch(s);
s* ← s1; k ← 1;
while k ≤ kmax do
    s2 ← Shake(s1,k);
    s3 ← LocalSearch(s2);
    s1, k ← AcceptanceCriterion(s*, s3)
s* ← LocalSearch(s*);
return s*
```

Algorithm 4 shows the pseudocode of our GVNS heuristic.

4.5 ILS-VND

Once a search procedure finds an optimal local solution. Throughout the ILS execution, the local search procedure is applied to the perturbation of the local optimal solutions previously found. So that local search is not executed for entirely new solutions. ILS's main idea is to focus on a reduced solution space instead of the entire solution space associated with the local optimal solutions [19]. The main methods of ILS are (1) $GenerateInitialSolution$, where a new solution is built, (2) $LocalSeach$, which improves the initial solution obtained; (3) $Perturb$, where a new solution is generated from the disturbance solution of the solution found by the

Algorithm 4: GVNS heuristic

```
s := FindInitialSolution(seed);
s_i := RandomMixShift(s) ;
diversification := 0 ;
for iter := 1 to MaxIter do
    perturbation phase ;
    s'_i := Shake(s_i) ;
    if f(s'_i) < f(s*) then
        s* := s'_i ;
        iter := 0;
    s_k := VND(s'_i);
    if f(s_k) < f(s*) then
        s* := s_k;
        iter := 0;
    else
        s_k :=AcceptanceCriteria(s*, s_k);
    s_i := s_k;
    diversification phase ;
    if iter = MaxIter and diversification = 0 then
        iter := 0;
        diversification := 1;
        s_i := Shake(s_i);
```

$LocalSearch$ method; (4) $AccetanceCriterion$, which determines which solutions should be considered to continue the search.

Algorithm 5 presents the pseudo-code of ILS-VND proposed for S-MDVRPPD. First, the algorithm initializes the $maxIter$ parameter. The number of iterations of the main loop is limited by $maxIter$. In each iteration, an initial solution is generated with the $GenerateInitialSolution$ procedure, the initial solution is improved using the $RandomMixShift$ procedure, and the $iterILS$ variable is fixed to 0. The number of iterations of the inner loop is also limited by $MaxIterILS$. Within the internal loop, the local search procedure (VND) is performed, generating a solution s. If the solution s is better than current solution s', then s' is updated with s, s is set with a perturbation of s' and the number of iterations $iterILS$ is set to 0. Otherwise, the number of iterations $iterILS$ is increased by one. Finally, if the current solution s' does not outperform the best solution $s*$, then we update $s*$ with s'.

5 Tabu Search

The tabu search (TS) explores part of the solution space by moving to the best neighbor of the current solution, even when this movement deteriorates the objective function [11]. Recently considered candidate solutions are stored in a structure

Algorithm 5: ILS-VND

```
for k := 1, ..., MaxIter do
    s := GenerateInitialSolution(seed);
    s' := RandomMixShift(s) ;
    iterILS := 0;
    while iterILS < MaxIterILS do
        r = number of neighborhoods ;
        s := VND(N(.), r, s) ;
        if f(s) < f(s') then
            s' := s ;
            s := Perturb(s') ;
            iterILS := 0 ;
        else
            iterILS := iterILS + 1 ;
    if f(s') < f(s*) then
        s* := s' ;
```

commonly known as a tabu list. To avoid cycling, solutions in a tabu list are made inaccessible for a number of iterations. To save time and memory space, it is stored only some attribute of the solution.

We present an adaptation of the TABUSTOCH heuristic proposed by Gendreau et al. [11] that was originally designed to the Vehicle Routing Problem (VRP) with stochastic demands. The algorithm solves a two stage stochastic VRP, where in the first stage a feasible solution is constructed including all vertices (clients). In the second stage recourse actions maybe taken, since the real demands of costumers are realized, capacity constraints may become violated. In traversing a route, once a vehicle becomes full it returns to the depot and resumes the route in the next client to be visited. All the parameters in the adapted algorithm, are the same used in the original TABUSTOCH. We will only present the modifications made to TABUSTOCH in order to deal with the S-MDVRPPD. Let x^k be a solution in the first stage in iteration k of the algorithm. Let $T(x^k)$ be the expected value in the second stage. Let $T(x^k) = \sum_{i=1}^{m^k} T^i(x^k)$, where m^k is the number of routes at iteration k.

The initial solution is built by assigning to each depot the closest *candidate* pair of pickup and delivery. A pair of pickup and delivery is a *candidate* if it has not been assigned to some depot. The selected pickup and delivery pair is appended to the solution (first pickup and then delivery). The initial solution is always feasible. The neighbourhood structure used by TABUSTOCH is the Mix-shift(1, 0) operator presented in Sect. 4.2. Thus, there is the possibility of inserting pairs of pickup and delivery in different cycles.

We consider the movement of nodes in a solution as elements of the tabu list. There are two ways to move a vertex: (1) change the position of the vertex in the same tour and (2) move the vertex (and its corresponding pair) to another tour. Either of these two movements is tabu for θ iterations, where θ is randomly selected from the

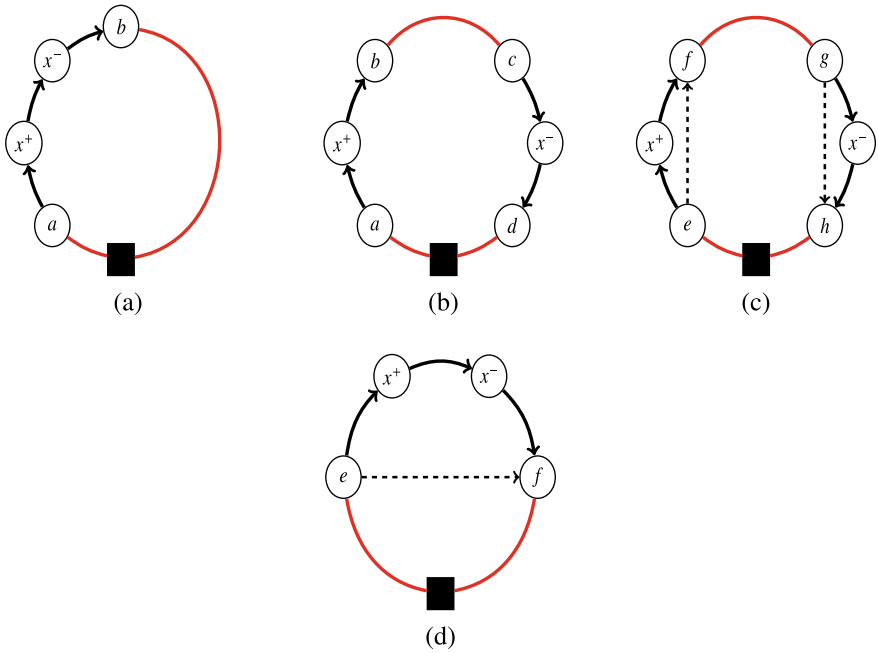


Fig. 9 Example of movements of nodes x^+ and x^-. Cases **a** and **b** represent different situations of nodes x^+ and x^- in a route, before the movement. Cases **c** and **d** represent possible insertion of x^+ and x^- in a route, after the movement

interval $[|V| - 5, |V|]$. The search of solutions in each iteration considers the current solution x^k and the best non-tabu solution x^{k+1} in the neighborhood structure Mix-Shift(1, 0). However, a tabu solution can be selected if it improves the best solution T^* (aspiration criteria).

Note that computing the expected value of a solution is expensive. Moving a pickup and delivery pair not only affects the cost related to their immediate neighbors, but also affects the cost of each node in the tour. Notice also that the movement affects the costs of both previous and new tours where they where inserted.

Suppose we wish to insert a pair of pickup and delivery $\{x^+, x^-\}$ into a route. Figure 9a and b represents the possible positions of $\{x^+, x^-\}$ in a route before removing them while Fig. 9c and d represents all possible positions of x^+ and x^- after their insertion into the new route. Dotted arrows represent arcs before insertion. Red lines represent subtours and black arrows arcs. We denote the approximations of the effect of inserting a pickup and delivery in a route with A_i and $\bar{A}_i$. A_i refers to the insertion of $\{x^+, x^-\}$ as shown in the Fig. 9c. $\bar{A}_i$ refers to the insertion of $\{x^+, x^-\}$ as shown in Fig. 9d. Note that the approximations of the effect of removing a pickup and delivery in a route can be represented with $-A_i$ and $-\bar{A}_i$.

We use three easy-computational approximations of insertion cost to speed up the search process. The first approximation, given by Eqs. (16) and (17), completely

detach the stochastic nature of the problem. The second approximation, given by Eqs. (18) and (19), partially remediate the first approximation, but these equations give all the weight for e, f, g, h and x^-. Taking into account P_e, P_f, P_g, P_h and P_{x^-}, the third approximation, given by Eqs. (20) and (21), seeks to remedy the second approximation. The problem with this last approximation happens when P_e, P_f, P_g, P_h are small and P_{x^+} (and so P_{x^-}) is large.

Tests conducted on 600 randomly generated instances involving between 10 and 100 vertices indicates that the third approximation yields the best correlation with the true cost increase ($r = 0.89$).

$$\begin{aligned} A_1(e, f, g, h, x^+, x^-) = {} & d_{ex^+} + d_{x^+f} + d_{gx^-} + d_{x^-h} \\ & - d_{ef} - d_{gh} \end{aligned} \tag{16}$$

$$\bar{A}_1(e, f, x^+, x^-) = d_{ex^+} + d_{x^+x^-} + d_{x^-f} - d_{ef} \tag{17}$$

$$\begin{aligned} A_2(e, f, g, h, x^+, x-) = {} & (d_{ex^+} + d_{x^+f} + d_{gx^-} + d_{x^-h} \\ & - d_{ef} - d_{gh})P_{x^+} \end{aligned} \tag{18}$$

$$\bar{A}_2(e, f, x^+, x^-) = (d_{ex^+} + d_{x^+x^-} + d_{x^-f} - d_{ef})P_{x^+} \tag{19}$$

$$\begin{aligned} A_3(e, f, g, h, x^+, x^-) = {} & d_{ex^+}P_eP_f + d_{x^+f}P_{x^+}P_f \\ & + d_{gx^-}P_gP_{x^-} + d_{x^-h}P_{x^-}P_h \\ & - d_{ef}P_eP_f - d_{gh}P_gP_h \end{aligned} \tag{20}$$

$$\begin{aligned} \bar{A}_3(e, f, x^+, x^-) = {} & d_{ex^+}P_eP_{x^+} + d_{x^+x^-}P_{x^+}P_{x^-} \\ & + d_{x^-f}P_{x^-}P_f - d_{ef}P_eP_f \end{aligned} \tag{21}$$

We have the necessary terms to *approximate the cost of a movement*. The expressions (22)–(25) are used to evaluate the movement cost of x^+ and x^-. We will use the cases shown in Fig. 3. If the movement of x^+ and x^- happens in the order: from case (Fig. 9a) to case (Fig. 9c) we use the Eq. (22); from case (Fig. 9a) to case (Fig. 9d) we use Eq. (23); from case (Fig. 9b) to case (Fig. 9c) we use (24); and, from case (Fig. 9b) to case (Fig. 9d) we use (25).

$$\Delta_1 = A_3(e, f, g, h, x^+, x^-) - \bar{A}_3(a, b, x^+, x^-) \tag{22}$$

$$\bar{\Delta}_1 = \bar{A}_3(e, f, x^+, x^-) - \bar{A}_3(a, b, x^+, x^-) \tag{23}$$

$$\Delta_2 = A_3(e, f, g, h, x^+, x^-) - A_3(a, b, c, d, x^+, x^-) \tag{24}$$

$$\bar{\Delta}_2 = \bar{A}_3(e, f, x^+, x^-) - A_3(a, b, c, d, x^+, x^-) \tag{25}$$

6 Computational Experiments

We conducted experiments using a data set derived from six TSPLIB instances (*ulysses16, bayg29, dantzig42, eil51, st70* and *st76*). For each of these instances, n vertices in the interval of [2, 10] were randomly selected to be depots. A random matching was performed among the other vertices to create pickup and delivery pairs. The probability of presence of each pickup and delivery pair was chosen uniformly in the interval [0, 1]. We generate 30 test instances.

The algorithms described above were coded in C++ and all experiments were run on a Linux operating system with 3 GB memory and Intel Core i5 2.54×4 Ghz processor. Computational times reported here are in CPU seconds on this machine. To evaluate the proposed algorithms (VNS and ILS-VND), we compare them with an adaptation of TABUSTOCH algorithm. Ten independent runs of the algorithms were performed for each test case.

The values of the parameters used by the algorithm ILS-VND were 10 and 15 for *MaxIter* and perturbation allowed *MaxIterILS*, respectively. The parameters α, $MaxIterShift$ and p were fixed to 1.05, 100 and 5, respectively. Regarding the parameterization of the VNS algorithm, the number of iterations *MaxIter* was 10. The parameters α, $MaxIterShift$ and p were fixed to 2.05, 100 and 6, respectively. All the parameters were calibrated empirically after preliminary tests with different values.

The algorithms' general performance was analysed with the averages of the values obtained by them in each instance. The Friedman test [10] was used to verify significant differences between the solutions obtained by the algorithms. This test was performed using the XLSTAT software [1]. For the interpretation of the test, we use H_0 (null hypothesis, samples come from the same population) and H_a (alternative hypothesis, samples do not come from the same population). The level of significance was $\alpha = 0.05$. If the calculated p-value is less than the significance level $\alpha = 0.05$, we must reject the null hypothesis H_0 and accept the alternative hypothesis H_a.

If the Friedman test rejects the null hypothesis, then we must execute the Nemenyi *post-hoc* procedure [22] to find concrete pairs that produce differences. If Nemenyi is executed, the correction of Bonferroni [8] must be used because there are multiple comparison in k groups.

After carrying out the experiments, Friedman's test found significant differences between the solutions obtained by the ILS-VND, VNS and Tabu Search algorithms, with a p-value 3.060200e−13. Since the calculated p-value is less than the significance level 0.05, we must reject the null hypothesis H_0 of equality of means and accept the alternative hypothesis H_a. The risk of rejecting the null hypothesis H_0, while it is true is less than 0.01%.

Table 2 shows the results of the ILS-VND, VNS, and TS algorithms. The best solutions are in boldface. The columns related to the instances show: the instance name, *Name*; the number of vertices in the graph, n; and the number of depots, $|D|$. The columns related to each algorithm are *Best*, and *Avg.* and *Time*. Columns *Avg.* and *Best* show the average and the best solution found by the algorithms in their ten

Table 2 Results of the algorithms for the instances of the S-MDVRPPD

Instance			TS			GVNS			ILS-VND		
Name	|V|	|D|	Best	Avg.	Time (s)	Best	Avg.	Time (s)	Best	Avg.	Time (s)
ulysses16a	16	2	69.96	71.14	0.46	67.20	67.21	0.72	**66.44**	67.41	0.91
ulysses16b	16	2	31.94	31.94	0.41	30.69	31.69	0.35	**30.68**	30.69	0.72
ulysses16c	16	2	45.26	45.93	0.66	45.78	45.78	0.63	**44.92**	44.92	0.70
ulysses16d	16	4	65.05	65.55	0.53	**57.59**	57.59	0.11	**57.59**	57.59	0.26
ulysses16e	16	4	81.09	81.09	0.60	**53.21**	54.34	0.13	**53.21**	53.21	0.27
bayg29a	29	3	8010.40	9348.96	11.37	7188.21	7726.93	5.33	**7082.80**	7247.16	6.30
bayg29b	29	9	15745.10	15745.10	8.50	11962.50	12001.30	0.18	**11890.90**	11990.20	0.46
bayg29c	29	9	13856.80	13856.80	7.43	10737.20	10802.60	0.17	**10631.60**	10660.90	0.61
bayg29d	29	3	7705.80	8449.30	10.00	7359.20	7462.77	5.62	**7121.43**	7219.10	8.34
bayg29e	29	5	10800.10	11700.60	11.27	8677.45	8723.20	2.59	**8078.92**	8116.32	4.37
dantzig42a	42	8	985.70	1009.46	28.37	723.32	741.81	2.73	**676.93**	696.76	5.11
dantzig42b	42	4	676.76	723.88	69.13	588.03	595.56	16.81	**542.06**	557.25	28.43
dantzig42c	42	10	1130.56	1130.56	35.98	747.50	762.20	1.85	**714.80**	730.77	2.00
dantzig42d	42	2	617.35	696.75	41.13	666.94	680.10	41.35	**599.89**	628.23	55.22
dantzig42e	42	10	1244.39	1334.65	45.51	738.07	752.72	3.08	**671.76**	680.96	3.41
eil51a	51	3	418.03	447.26	122.13	367.04	372.13	104.90	**351.98**	369.15	90.39
eil51b	51	7	537.55	586.47	66.46	409.61	418.76	18.07	**368.78**	384.45	24.70

(continued)

Table 2 (continued)

Instance			TS			GVNS			ILS-VND		
Name	|V|	|D|	Best	Avg.	Time (s)	Best	Avg.	Time (s)	Best	Avg.	Time (s)
eil51c	51	9	579.64	596.45	45.63	362.87	370.42	9.13	**340.74**	354.02	11.94
eil51d	51	7	493.12	535.91	107.50	339.97	342.45	52.47	**323.60**	329.68	39.62
eil51e	51	5	409.80	484.96	93.16	350.28	352.09	26.00	**339.41**	344.22	22.97
st70a	70	4	810.95	875.85	408.45	695.13	710.47	179.06	**621.25**	690.76	224.80
st70b	70	6	871.67	945.15	765.90	796.82	804.33	66.16	**728.87**	760.81	88.89
st70c	70	8	1078.96	1249.14	459.02	726.94	746.84	84.25	**680.43**	710.06	115.43
st70d	70	8	1012.48	1071.10	398.99	687.31	714.88	91.39	**620.63**	669.17	87.50
st70e	70	6	960.09	1048.68	483.40	645.59	665.95	161.70	**575.27**	616.96	200.63
eil76a	76	6	703.79	790.67	480.91	590.74	606.13	159.47	**556.89**	599.37	171.47
eil76b	76	2	491.96	530.28	352.76	490.33	493.89	499.53	**464.73**	486.68	606.41
eil76c	76	4	559.31	636.99	381.38	581.92	599.15	272.26	**513.01**	564.59	309.40
eil76d	76	6	694.46	806.43	582.45	557.33	583.57	171.79	**522.01**	557.11	236.04
eil76e	76	6	642.77	723.63	404.50	568.11	581.04	158.99	**490.29**	534.55	176.54

Table 3 Nemenyi p-values between pairs of algorithms

	TS	GVNS	ILS-VND
TS	1	2.229e-29	8.360e-44
GVNS	2.229e-29	1	2.850e-26
ILS-VND	8.360e-44	2.850e-26	1

independent executions, respectively. Column *Time* presents the average processing time, in seconds, spend by each algorithm. The ILS-VND presented the best results for all instances.

A Table 3 presents the p-values between pairs of algorithms. Significant differences are shown between the all the algorithms, given that the its p-values are less than the significance level of 0.05. Therefore all the algorithms belong to different samples. The results show a clear superiority of ILS-VND compared to GVNS and TS, finding equal or better solutions in all instances.

7 Conclusion

This article described a new and practical SVRP involving multiple depots, and pickup and delivery (S-MDVRPPD). Contrary to the deterministic case, it is not easy to compute the objective function associated with a solution [4]. We presented a closed-form expression to compute the expected length of an a priori sequence under general probabilistic assumptions. In order to deal with S-MDVRPPD, two algorithms based on the Iterated Local Search and VNS metaheuristics were proposed, which use a VND heuristic as a local search procedure. We use six local search operators, *Shift(1, 0), Swap(1, 1), 2-opt, 3-opt, Reverse, and Mix-Shift*. Also, we use two perturbation mechanisms, *Double-Swap and Depot-exchange*. We propose a heuristic based on the *Mix-shift* operator to refine the initial solution of the VNS and ILS-VND. The VNS and ILS-VND were compared with a tabu search algorithm (TABUSTOCH). We report the results for 30 instances. The results show that the ILS-VND was superior for all instances tested. Our approach can be used as benchmark for future research in this area. The S-MDVRPPD can be further generalized to handle more practical constraints, e.g., limited capacity vehicles, time windows and stochastic demands.

References

1. Addinsoft: Data Analysis and Statistical Solution for Microsoft Excel, Paris, France (2017)
2. Assaf, M., Ndiaye, M.: Multi travelling salesman problem formulation. In: 2017 4th International Conference on Industrial Engineering and Applications (ICIEA). IEEE (2017). https://doi.org/10.1109/iea.2017.7939224

3. Bektas, T.: The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega **34**(3), 209–219 (2006). https://doi.org/10.1016/j.omega.2004.10.004
4. Bertsimas, D.J.: A vehicle routing problem with stochastic demand. Oper. Res. **40**(3), 574–585 (1992). https://doi.org/10.1287/opre.40.3.574
5. Carrabs, F., Cordeau, J.-F., Laporte, G.: Variable neighborhood search for the pickup and delivery traveling salesman problem with LIFO loading. INFORMS J. Comput. **19**(4), 618–632 (2007). https://doi.org/10.1287/ijoc.1060.0202
6. Crevier, B., Cordeau, J.-F., Laporte, G.: The multi-depot vehicle routing problem with inter-depot routes. Europ. J. Oper. Res. **176**(2), 756–773 (2007). https://doi.org/10.1016/j.ejor.2005.08.015
7. Dridi, I.H., et al.: Optimisation of the multi-depots pick-up and delivery problems with time windows and multi-vehicles using PSO algorithm. Int. J. Prod. Res. 1–14 (2019). https://doi.org/10.1080/00207543.2019.1650975
8. Dunn, O.J.: Multiple comparisons among means. J. Amer. Stat. Ass. **56**(293), 52–64 (1961)
9. Erera, A.L., Savelsbergh, M., Uyar, E.: Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. Networks **54**(4), 270–283 (2009). https://doi.org/10.1002/net.20338
10. Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Amer. Stat. Ass. **32**(200), 675–701 (1937)
11. Gendreau, M., Laporte, G., Séguin, R.: A Tabu search heuristic for the vehicle routing problem with stochastic demands and customers. Oper. Res. **44**(3), 469–477 (1996). https://doi.org/10.1287/opre.44.3.469
12. Goodson, J.C.: A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands. Europ. J. Oper. Res. **241**(2), 361–369 (2015). https://doi.org/10.1016/j.ejor.2014.09.031
13. Hansen, P., Mladenovic, N.: Variable neighborhood search. In: Search Methodologies, pp. 211–238. Springer US (2005). https://doi.org/10.1007/0-387-28356-0_8
14. Hansen, P., et al.: Variable neighborhood search: basics and variants. EURO J. Comput. Optim. **5**(3), 423–454 (2016). https://doi.org/10.1007/s13675-016-0075-x
15. Kachitvichyanukul, V., Sombuntham, P., Kunnapapdeelert, S.: Two solution representations for solving multi-depot vehicle routing problem with multiple pickup and delivery requests via PSO. Comput. & Ind. Eng. **89**, 125–136 (2015). https://doi.org/10.1016/j.cie.2015.04.011
16. Kara, I., Bektas, T.: Integer linear programming formulations of multiple salesman problems and its variations. Europ. J. Oper. Res. **174**(3), 1449–1458 (2006). https://doi.org/10.1016/j.ejor.2005.03.008
17. Kuo, Y., Wang, C.-C.: A variable neighborhood search for the multi-depot vehicle routing problem with loading cost. Expert Syst. Appl. **39**(8), 6949–6954 (2012). https://doi.org/10.1016/j.eswa.2012.01.024
18. Laporte, G., Musmanno, R., Vocaturo, F.: An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. Trans. Sci. **44**(1), 125–135 (2010). https://doi.org/10.1287/trsc.1090.0290
19. Lourenço, H.R., Martin, O., St ützle, T.: Iterated local search. In: Handbook of Metaheuristics, pp. 321–353 (2003)
20. Mendoza, J.E., Villegas, J.G.: A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Optim. Lett. **7**(7), 1503–1516 (2012). https://doi.org/10.1007/s11590-012-0555-8
21. Mladenovic, N., Hansen, P.: Variable neighborhood search. Comput. & Oper. Res. **24**(11), 1097–1100 (1997). https://doi.org/10.1016/s0305-0548(97)00031-2
22. Nemenyi, P.: Distribution-free multiple comparisons. Biometrics **18**(2), 263 (1962). International Biometric Soc 1441 I ST, NW, Suite 700, Washington, DC 20005-2210
23. Oyola, J., Arntzen, H., Woodruff, D.L.: The stochastic vehicle routing problem, a literature review, Part II: solution methods. EURO J. Trans. Log. **6**(4), 349–388 (2016). https://doi.org/10.1007/s13676-016-0099-7

24. Pereira, V.N.G., et al. (2018) The Steiner multi cycle problem with applications to a collaborative truckload problem. In: 17th International Symposium on Experimental Algorithms (SEA 2018). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018). https://doi.org/10.4230/LIPICS.SEA.2018.26
25. Polacek, M., et al.: A variable neighborhood search for the multi depot vehicle routing problem with time windows. J. Heuristics **10**(6), 613–627 (2004). https://doi.org/10.1007/s10732-005-5432-5
26. Pongchairerks, P.: Forward VNS, reverse VNS, and multi-VNS algorithms for job-shop scheduling problem. Modell. Simul. Eng. **2016**, 1–15 (2016). https://doi.org/10.1155/2016/5071654
27. Psaraftis, H.N., Wen, M., Kontovas, C.A.: Dynamic vehicle routing problems: three decades and counting. Networks **67**(1), 3–31 (2015). https://doi.org/10.1002/net.21628
28. Rios, B.H.O., et al.: Stochastic multi-depot vehicle routing problem with pickup and delivery: an ILS approach. In: Proceedings of the 2020 Federated Conference on Computer Science and Information Systems. IEEE (2020). https://doi.org/10.15439/2020f127
29. Rios, B.H.O., Goldbarg, E.F.G., Goldbarg, M.C.: A hybrid metaheuristic for the traveling car renter salesman problem. In: 2017 Brazilian Conference on Intelligent Systems (BRACIS). IEEE (2017). https://doi.org/10.1109/bracis.2017.20
30. Rios, B.H.O., Goldbarg, E.F.G., Quesquen, G.Y.O.: A hybrid metaheuristic using a corrected formulation for the Traveling Car Renter Salesman Problem. In: 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE (2017). https://doi.org/10.1109/cec.2017.7969584
31. Stodola, P.: Hybrid ant colony optimization algorithm applied to the multi-depot vehicle routing problem. Natural Comput. **19**(2), 463–475 (2020). https://doi.org/10.1007/s11047-020-09783-6

Evaluation of MO-ACO Algorithms Using a New Fast Inter-Criteria Analysis Method

Jean Dezert, Stefka Fidanova, and Albena Tchamova

Abstract In this paper, we present a fast Belief Function based Inter-Criteria Analysis (BF-ICrA) method based on the canonical decomposition of basic belief assignments defined on a dichotomous frame of discernment. This new method is then applied for evaluating the Multiple-Objective Ant Colony Optimization (MO-ACO) algorithm for Wireless Sensor Networks (WSN) deployment and for Workforce Planning Problem (WPP).

Keywords Inter-Criteria Analysis · Ant Colony Optimization · Belief functions · Canonical decomposition · Proportional Conflict Redistribution · Fusion rules

1 Introduction

In our previous work [1] we propose a new and improved version of classical Atanassov's InterCriteria Analysis (ICrA) [2–4] approach based on Belief Functions (BF-ICrA). This method proposes a better construction of Inter-Criteria Matrix that fully exploits all the information of the score matrix, and the closeness measure of agreement between criteria based on belief interval distance. In [6], we show how the fusion of many sources of evidences represented by Basic Belief Assignments (BBAs) defined on a same dichotomous frame of discernment can be fast and easily done thanks to the Proportional Conflict Redistribution rule no. 5 based canonical decomposition of the BBAs, proposed recently in [7]. In [8] we did show how to use this fast fusion method for decision-making support. In this paper we propose

J. Dezert
The Frenche Aerospace Lab/ONERA, Palaiseau, France
e-mail: jean.dezert@onera.fr

S. Fidanova (✉) · A. Tchamova
Institute of I&C Technology, Bulgarian Academy of Sciences, Sofia, Bulgaria
e-mail: stefka@parallel.bas.bg

A. Tchamova
e-mail: tchamova@bas.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_3

a new fast BF-ICrA method based on this canonical decomposition. Then we show how to apply it for the evaluation of the Multiple-Objective Ant Colony Optimization (MO-ACO) algorithm for Wireless Sensor Networks (WSN) deployment, and for workforce planning problem. After a condensed presentation of basics of belief functions in Sect. 2, including the short description of canonical decomposition of dichotomous BBAs approach (in Sect. 2.2), and the main steps of fast fusion method of dichotomous BBAs (in Sect. 2.3), the BF-ICrA method is described and analyzed in Sect. 3, and the fast BF-ICrA method in Sect. 4. In the Sect. 5 we present the Multiple-Objective Ant Colony Optimization (MO-ACO) algorithm. In Sect. 6 the results of the fast BF-ICrA method with the MO-ACO algorithm for WSN layout deployment [22] are presented and discussed. The fast BF-ICrA method has also been applied to the workforce planning problem [24], and the results are presented in Sect. 7. Conclusion is given in Sect. 8.

2 Basics of Belief Functions

2.1 Basic Definitions

Belief functions (BF) have been introduced by Shafer in [9] to model epistemic uncertainty and to combine distinct sources of evidence thanks to Dempster's rule of combination. In Shafer's framework, we assume that the answer[1] of the problem under concern belongs to a known finite discrete frame of discernment (FoD) $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$, with $n > 1$, and where all elements of Θ are mutually exclusive and exhaustive. The set of all subsets of Θ (including empty set $\emptyset$ and Θ) is the power-set of Θ denoted by 2^Θ. A proper Basic Belief Assignment (BBA) associated with a given source of evidence is defined [9] as a mapping $m(\cdot) : 2^\Theta \to [0, 1]$ satisfying $m(\emptyset) = 0$ and $\sum_{A \in 2^\Theta} m(A) = 1$. The quantity $m(A)$ is called the mass of A committed by the source of evidence. Belief and plausibility functions are respectively defined from a proper BBA $m(\cdot)$ by

$$Bel(A) = \sum_{B \in 2^\Theta | B \subseteq A} m(B) \tag{1}$$

and

$$Pl(A) = \sum_{B \in 2^\Theta | A \cap B \neq \emptyset} m(B) = 1 - \mathrm{Bel}(\bar{A}). \tag{2}$$

where $\bar{A}$ is the complement of A in Θ.

$Bel(A)$ and $Pl(A)$ are usually interpreted respectively as lower and upper bounds of an unknown (subjective) probability measure $P(A)$. The quantities $m(\cdot)$ and $Bel(\cdot)$

[1] I.e. the solution, or the decision to take.

are one-to-one and linked by the Möbius inverse formula (see [9], p. 39). A is called a Focal Element (FE) of $m(\cdot)$ if $m(A) > 0$. When all focal elements are singletons, $m(\cdot)$ is called a *Bayesian BBA* [9] and its corresponding $Bel(\cdot)$ function is equal to $Pl(\cdot)$ and they are homogeneous to a (subjective) probability measure $P(\cdot)$. The vacuous BBA, representing a totally ignorant source, is defined as $m_v(\Theta) = 1$. A dichotomous BBA is a BBA defined on a FoD which has only two proper subsets, for instance $\Theta = \{A, \bar{A}\}$ with $A \neq \Theta$ and $A \neq \emptyset$. A dogmatic BBA is a BBA such that $m(\Theta) = 0$. If $m(\Theta) > 0$ the BBA $m(\cdot)$ is nondogmatic. A simple BBA is a BBA that has at most two focal sets and one of them is Θ. A dichotomous non dogmatic mass of belief is a BBA having three focal elements A, $\bar{A}$ and $A \cup \bar{A}$ with A and $\bar{A}$ subsets of Θ.

In his Mathematical Theory of Evidence [9], Shafer proposed to combine $s \geq 2$ distinct sources of evidence represented by BBAs with Dempster's rule (i.e. the normalized conjunctive rule), which unfortunately behaves counterintuitively both in high and low conflicting situations as reported in [10–13]. In our previous works (see [14], Vols. 2 and 3 for full justification and examples) we did propose new rules of combination based on different Proportional Conflict Redistribution (PCR) principles, and we have shown the interest of the PCR rule No 5 (PCR5) for combining two BBAs, and PCR rule No 6 (PCR6) for combining more than two BBAs altogether [14], Vol. 2. PCR6 coincides with PCR5 when one combines two sources. The difference between PCR5 and PCR6 lies in the way the proportional conflict redistribution is done as soon as three (or more) sources are involved in the fusion. PCR5 transfers the conflicting mass only to the elements involved in the conflict and proportionally to their individual masses, so that the specificity of the information is entirely preserved in this fusion process.

The general (complicate) formulas for PCR5 and PCR6 rules are given in [14], Vol. 2. The fusion of two BBAs based on PCR5 (or PCR6) rule which will be use for canonical decomposition of a dichotomous BBA is obtained by the formula

$$m_{PCR5}(X) = \sum_{\substack{X_1, X_2 \in 2^\Theta \\ X_1 \cap X_2 = X}} m_1(X_1) m_2(X_2) + \sum_{\substack{X_2 \in 2^\Theta \\ X_2 \cap X = \emptyset}} \left[\frac{m_1(X)^2 m_2(X_2)}{m_1(X) + m_2(X_2)} + \frac{m_2(X)^2 m_1(X_2)}{m_2(X) + m_1(X_2)} \right] \quad (3)$$

where all denominators in (3) are different from zero. If a denominator is zero, that fraction is discarded.

From the implementation point of view, PCR6 is simpler to implement than PCR5. For convenience, very basic (not optimized) Matlab™ codes of PCR5 and PCR6 fusion rules can be found in [14, 15] and from the toolboxes repository on the web [16]. The main drawback of PCR5 and PCR6 rules is their very high combinatorial complexity when the number of source is big, as well as the cardinality of the FoD. In this case, PCR5 or PCR6 rules cannot be used directly because of memory overflow.

Even for combining BBAs defined on a simple dichotomous FoD as those involved in the Inter-Criteria Analysis (ICrA), the computational time for combining more than 10 sources can take several hours.[2] That is why a fast fusion method to combine dichotomous BBAs is necessary, and we present it in the next subsections.

2.2 Canonical Decomposition of Dichotomous BBA

A FoD $\Theta = \{A, \bar{A}\}$ is called dichotomous if it consists of only two proper subsets A and $\bar{A}$ with $A \cup \bar{A} = \Theta$ and $A \cap \bar{A} = \emptyset$, where $\bar{A}$ is the complement of A in Θ and A is different from Θ and from Empty-Set. We consider a given proper BBA $m(\cdot) : 2^{\Theta} \rightarrow [0, 1]$ of the general form

$$m(A) = a, \quad m(\bar{A}) = b, \quad m(A \cup \bar{A}) = 1 - a - b \tag{4}$$

The canonical decomposition problem consists in finding the two following simple proper BBAs m_p and m_c of the form

$$m_p(A) = x, \quad m_p(A \cup \bar{A}) = 1 - x \tag{5}$$

$$m_c(\bar{A}) = y, \quad m_c(A \cup \bar{A}) = 1 - y \tag{6}$$

with $(x, y) \in [0, 1] \times [0, 1]$, such that $m = Fusion(m_p, m_c)$, for a chosen rule of combination denoted by $Fusion(\cdot, \cdot)$. The simple BBA $m_p(\cdot)$ is called the *pro-BBA* (or pro-evidence) of A, and the simple BBA $m_c(\cdot)$ the *contra-BBA* (or contra-evidence) of A. The BBA $m_p(\cdot)$ is interpreted as a source of evidence providing an uncertain evidence in favor of A, whereas $m_c(\cdot)$ is interpreted as a source of evidence providing an uncertain contrary evidence about A.

In [7], we have shown that this decomposition is possible with Dempster's rule only if $0 < a < 1$, $0 < b < 1$ and $a + b < 1$, and we have $x = \frac{a}{1-b}$ and $y = \frac{b}{1-a}$. However, any dogmatic BBA $m(A) = a, m(\bar{A}) = b$ with $a + b = 1$ is not decomposable from Dempster's rule for the case when $(a, b) \neq (1, 0)$ and $(a, b) \neq (0, 1)$, and the dogmatic BBAs $m(A) = 1$, $m(\bar{A}) = 0$, or $m(A) = 0$, $m(\bar{A}) = 1$ have infinitely many decompositions based on Dempster's rule of combination. We have also proved that this canonical decomposition cannot be done from conjunctive, disjunctive, Yager's [17] or Dubois-Prade [18] rules of combination, neither from the averaging rule. The main result of [7] is that this canonical decomposition is unique and is always possible in all cases using the PCR5 rule of combination. This is very useful to implement a fast efficient approximating fusion method of dichotomous BBAs as presented in details in [6]. We recall the following two important theorems proved in [7].

[2] With a MacBook Pro 2.8 GHz Intel Core i7 with 16 Go 1600 MHz DDR3 memory running Matlab™ R2018a.

Theorem 1 *Consider a dichotomous FoD* $\Theta = \{A, \bar{A}\}$ *with* $A \neq \Theta$ *and* $A \neq \emptyset$ *and a nondogmatic BBA* $m(\cdot) : 2^{\Theta} \rightarrow [0, 1]$ *defined on* Θ *by* $m(A) = a$, $m(\bar{A}) = b$, *and* $m(A \cup \bar{A}) = 1 - a - b$, *where* $a, b \in [0, 1]$ *and* $a + b < 1$. *Then the BBA* $m(\cdot)$ *has a unique canonical decomposition using PCR5 rule of combination of the form* $m = PCR5(m_p, m_c)$ *with pro-evidence* $m_p(A) = x$, $m_p(A \cup \bar{A}) = 1 - x$ *and contra-evidence* $m_c(\bar{A}) = y$, $m_c(A \cup \bar{A}) = 1 - y$ *where* $x, y \in [0, 1]$.

Theorem 2 *Any dogmatic BBA defined by* $m(A) = a$ *and* $m(\bar{A}) = b$, *where* $a, b \in [0, 1]$ *and* $a + b = 1$, *has a canonical decomposition using PCR5 rule of combination of the form* $m = PCR5(m_p, m_c)$ *with* $m_p(A) = x$, $m_p(A \cup \bar{A}) = 1 - x$ *and* $m_c(\bar{A}) = y$, $m_c(A \cup \bar{A}) = 1 - y$ *where* $x, y \in [0, 1]$.

Theorems 1 and 2 prove that the decomposition based on PCR5 always exists and it is unique for any dichotomous (nondogmatic, or dogmatic) BBA.

For the case of dichotomous nondogmatic BBA considered in Theorem 1, one has to find x and y solutions of the system

$$a = x(1 - y) + \frac{x^2 y}{x + y} = \frac{x^2 + xy - xy^2}{x + y} \tag{7}$$

$$b = (1 - x)y + \frac{xy^2}{x + y} = \frac{y^2 + xy - x^2 y}{x + y} \tag{8}$$

under the constraints $(a, b) \in [0, 1]^2$, and $0 < a + b < 1$. The explicit expression of x and y are difficult to obtain analytically (even with modern symbolic computing systems like Mathematica™, or Maple™) because one has a quartic equation to solve whose general analytical expression of its solutions is very complicate. Fortunately, the solutions can be easily calculated numerically by these computing systems, and even with Matlab™ system (thanks to the *fsolve* function) as soon as the numerical values are committed to a and to b, and this is what we use in our simulations.

2.3 Fast Fusion of Dichotomous BBAs

The main idea for making the fast fusion of dichotomous BBAs $m_s(.)$, for $s = 1, 2, \ldots, S$ defined on the same FoD Θ is based on the three following main steps:

1. In the first step, one decomposes canonically each dichotomous BBA $m_s(\cdot)$ into its pro and contra evidences $m_{p,s} = (m_{p,s}(A), m_{p,s}(\bar{A}), m_{p,s}(A \cup \bar{A})) = (x_s, 0, 1 - x_s)$ and $m_{c,s} = (m_{c,s}(A), m_{c,s}(\bar{A}), m_{c,s}(A \cup \bar{A})) = (0, y_s, 1 - y_s)$,
2. In the second step, one combines the pro-evidences $m_{p,s}$ for $s = 1, 2, \ldots, S$ altogether to get a global pro-evidence m_p, and in parallel one combines all the contra-evidences $m_{c,s}$ for $s = 1, 2, \ldots, S$ altogether to get a global contra-evidence m_c. The fusion step of pro and contra evidences is based on conjunctive rule of combination.

3. Once m_p and m_c are calculated, then one combines them with PCR5 fusion rule to get the final result.

Because the PCR5 rule of combination is not associative, the fusion of the canonical BBAs followed by their PCR5 fusion will not provide in general the same result as the direct fusion of the dichotomous BBAs altogether but only an approximate result, which is normal. However, this new fusion approach is interesting because the fusion of the pro-evidence $m_{p,s}$ (resp. contra-evidences $m_{c,s}$) is very simple because there is non conflict between $m_{p,s}$ (resp. between $m_{c,s}$), so that their fusion can be done quite easily and a large number of sources can be combined without a high computational burden. In fact, with this fusion approach, only one PCR5 fusion step of simple (combined) canonical BBAs is needed at the very end of the fusion process. In [6], we have proved with a Monte-Carlo simulation analysis that the approximation obtained by this new fusion method based on the fusion of pro-evidences and contra-evidences with respect to the direct fusion of the BBAs with PCR5 (or PCR6 when considering more than two sources to combine) is effective because the agreement between the decision taken from the direct fusion method, and the indirect (canonical decomposition based) method is very good. This new fusion method based on this canonical decomposition does not suffer of combinatorial complexity limitation which is of great interest in some applications because many (hundreds or even thousands) of dichotomous BBAs could be easily combined very quickly. Actually with this method what takes a bit time is only the canonical decomposition done by the numerical solver. Our analysis [6] has shown that complexity of this fast approach is quasi-linear with the number of sources to combine.

3 The BF-ICrA Method

In [1], we did present an improved version of Atanassov's Inter-Criteria Analysis (ICrA) method [2–4] based on belief functions. This new method has been named BF-ICrA (Belief Function based Inter-Criteria Analysis) for short. It has already been applied to GPS surveying problems in [19]. We present briefly in this section the principles of BF-ICrA.

BF-ICrA starts with the construction of an $M \times N$ BBA matrix $\mathbf{M} = [m_{ij}(\cdot)]$ from the score matrix $\mathbf{S} = [S_{ij}]$. The BBA matrix $\mathbf{M}$ is obtained as follows—see [20] for details and justification.

$$m_{ij}(A_i) = Bel_{ij}(A_i) \tag{9}$$

$$m_{ij}(\bar{A}_i) = Bel_{ij}(\bar{A}_i) = 1 - Pl_{ij}(A_i) \tag{10}$$

$$m_{ij}(A_i \cup \bar{A}_i) = Pl_{ij}(A_i) - Bel_{ij}(A_i) \tag{11}$$

where[3]

$$Bel_{ij}(A_i) \triangleq Sup_j(A_i)/A^j_{\max} \tag{12}$$

$$Bel_{ij}(\bar{A}_i) \triangleq Inf_j(A_i)/A^j_{\min} \tag{13}$$

with

$$Sup_j(A_i) \triangleq \sum_{k\in\{1,\ldots M\}|S_{kj}\leq S_{ij}} |S_{ij} - S_{kj}| \tag{14}$$

$$Inf_j(A_i) \triangleq -\sum_{k\in\{1,\ldots M\}|S_{kj}\geq S_{ij}} |S_{ij} - S_{kj}| \tag{15}$$

and

$$A^j_{\max} \triangleq \max_i Sup_j(A_i) \tag{16}$$

$$A^j_{\min} \triangleq \min_i Inf_j(A_i) \tag{17}$$

For another criterion $C_{j'}$ and the j'th column of the score matrix we will obtain another set of BBA values $m_{ij'}(\cdot)$. Applying this method for each column of the score matrix we are able to compute the BBA matrix $\mathbf{M} = [m_{ij}(\cdot)]$ whose each component is in fact a triplet $(m_{ij}(A_i), m_{ij}(\bar{A}_i), m_{ij}(A_i \cup \bar{A}_i))$ of BBA values in [0, 1] such that $m_{ij}(A_i) + m_{ij}(\bar{A}_i) + m_{ij}(A_i \cup \bar{A}_i)) = 1$ for all $i = 1, \ldots, M$ and $j = 1, \ldots, N$.

The next step of BF-ICrA approach is the construction of the $N \times N$ Inter-Criteria Matrix $\mathbf{K} = [K_{jj'}]$ from $M \times N$ BBA matrix $\mathbf{M} = [m_{ij}(\cdot)]$ where elements $K_{jj'}$ corresponds to the BBA $(m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))$ about positive consonance θ, negative consonance $\bar{\theta}$ and uncertainty between criteria C_j and $C_{j'}$ respectively. The construction of the triplet $K_{jj'} = (m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))$ is based on two steps:

- **Step 1 (BBA construction)**: Getting $m^i_{jj'}(.)$.
  For each alternative A_i for $i = 1, \ldots, M$, we first compute the belief assignment $(m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}))$ for any two criteria $j, j' \in \{1, 2, \ldots, N\}$. For this, we consider two sources of evidences (SoE) indexed by j and j' providing the BBA m_{ij} and $m_{ij'}$ defined on the simple FoD $\{A_i, \bar{A}_i\}$ and denoted $m_{ij} = [m_{ij}(A_i), m_{ij}(\bar{A}_i), m_{ij}(A_i \cup \bar{A}_i)]$ and $m_{ij'} = [m_{ij'}(A_i), m_{ij'}(\bar{A}_i), m_{ij'}(A_i \cup \bar{A}_i)]$. We also denote $\Theta = \{\theta, \bar{\theta}\}$ the FoD about the relative state of the two SoE, where θ means that the two SoE agree, $\bar{\theta}$ means that they disagree and $\theta \cup \bar{\theta}$ means that we don't know. Hence, two SoE are in total agreement if both commit their maximum belief mass to the same element A_i or to the same element $\bar{A}_i$. Similarly, two SoE are in total disagreement if each one commits its maximum mass of belief to one element and the other to its opposite, that is if one has $m_{ij}(A_i) = 1$ and

[3] Assuming that $A^j_{\max} \neq 0$ and $A^j_{\min} \neq 0$. If $A^j_{\max} = 0$ then $Bel_{ij}(A_i) = 0$, and if $A^j_{\min} = 0$ then $Pl_{ij}(A_i) = 1$.

$m_{ij'}(\bar{A}_i) = 1$, or if $m_{ij}(\bar{A}_i) = 1$ and $m_{ij'}(A_i) = 1$. Based on this very simple and natural principle, one can now compute the belief masses as follows:

$$m^i_{jj'}(\theta) = m_{ij}(A_i)m_{ij'}(A_i) + m_{ij}(\bar{A})m_{ij'}(\bar{A}) \tag{18}$$

$$m^i_{jj'}(\bar{\theta}) = m_{ij}(A_i)m_{ij'}(\bar{A}_i) + m_{ij}(\bar{A}_i)m_{ij'}(A_i) \tag{19}$$

$$m^i_{jj'}(\theta \cup \bar{\theta}) = 1 - m^i_{jj'}(\theta) - m^i_{jj'}(\bar{\theta}) \tag{20}$$

$m^i_{jj'}(\theta)$ represents the degree of agreement between the BBA $m_{ij}(\cdot)$ and $m_{ij'}(\cdot)$ for the alternative A_i, $m^i_{jj'}(\bar{\theta})$ represents the degree of disagreement of the two BBAs and $m^i_{jj'}(\theta \cup \bar{\theta})$ the level of uncertainty (i.e. how much we don't know if they agree or disagree). By construction $m^i_{jj'}(\cdot) = m^i_{j'j}(\cdot)$, $m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}) \in [0, 1]$ and $m^i_{jj'}(\theta) + m^i_{jj'}(\bar{\theta}) + m^i_{jj'}(\theta \cup \bar{\theta}) = 1$. This BBA modeling permits to build a set of M symmetrical Inter-Criteria Belief Matrices (ICBM) $\mathbf{K}^i = [K^i_{jj'}]$ of dimension $N \times N$ relative to each alternative A_i whose components $K^i_{jj'}$ correspond to the triplet of BBA values $m^i_{jj'} = (m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}))$ modeling the belief of agreement and of disagreement between C_j and $C_{j'}$ based on A_i.

- **Step 2 (fusion)**: Getting $\mathbf{m_{jj'}}(.)$.
  In this step, one needs to combine the BBAs $\mathbf{m^i_{jj'}}(.)$ for $i = 1, \ldots, M$ altogether to get the component $K_{jj'} = (m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))$ of the Inter-Criteria Belief matrix[4] (ICBM) $\mathbf{K} = [K_{jj'}]$. For this and from the theoretical standpoint, we recommend to use the PCR6 fusion rule [14] (Vol. 3) because of known deficiencies of Dempster's rule.

Once the global Inter-Criteria Belief Matrix (ICBM) is calculated as the matrix $\mathbf{K} = [K_{jj'} = (m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))]$, we can identify the criteria that are in strong agreement, in strong disagreement, and those on which we are uncertain. For identifying the criteria that are in strong agreement, we evaluate the distance of each component of $K_{jj'}$ with the BBA representing the best agreement state and characterized by the specific BBA[5] $m_T(\theta) = 1$. From a similar approach we can also identify, if we want, the criteria that are in very strong disagreement using the distance of $m_{jj'}(\cdot)$ with respect to the BBA representing the best disagreement state characterized by the specific BBA $m_F(\bar{\theta}) = 1$. We use the belief interval distance $d_{BI}(m_1, m_2)$ presented in [21] for measuring the distance between the two BBAs.

[4] For presentation convenience, the ICBM $\mathbf{K} = [K_{jj'} = (m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))]$ is decomposed into three matrices $\mathbf{K}(\theta) = [K^{\theta}_{jj'} = m_{jj'}(\theta)]$, $\mathbf{K}(\bar{\theta}) = [K^{\bar{\theta}}_{jj'} = m_{jj'}(\bar{\theta})]$ and $\mathbf{K}(\theta \cup \bar{\theta}) = [K^{\theta \cup \bar{\theta}}_{jj'} = 1 - m_{jj'}(\theta) - m_{jj'}(\bar{\theta})]$.

[5] We use the index T in the notation $m_T(\cdot)$ to refer that the agreement is true, and F in $m_F(\cdot)$ to specify that the agreement is false.

4 Fast BF-ICrA Method

The computational complexity of BF-ICrA is of course higher than the complexity of ICrA because it makes a more precise evaluation of local and global inter-criteria belief matrices with respect to inter-criteria matrices calculated by Atanassov's ICrA. The overall reduction of the computational burden of the original MCDM problem thanks to BF-ICrA depends highly on the problem under concern, the complexity and cost to evaluate each criteria involved in it, as well as the number of redundant criteria identified by BF-ICrA method.

The main drawback of BF-ICrA method is the PCR6 combination required in its step 2 for combining altogether the dichotomous BBAs $m^i_{jj'}(.)$. Because of combinatorial complexity of PCR6 rule, it cannot work in reasonable computational time as soon as the number of sources to combine altogether is greater than 10, which prevents its use for solving ICrA problems involving more than 10 alternatives (as in the Examples 2 and 3 presented in Sect. 6). That is why it is necessary to adapt the original BF-ICrA method for working with a large number of alternatives and criteria. For this, we can in step 2 of BF-ICrA exploit the method for the fast fusion of dichotomous BBAs presented in Sect. 2.3. More precisely, each dichotomous BBA $m^i_{jj'}(.)$ will be canonically decomposed in its pro-evidence $m^i_{jj',p}(.)$ and its contra-evidence $m^i_{jj',c}(.)$ that will be combined separately to get the global pro-evidence $m_{jj',p}(.)$ and the global contra-evidence $m_{jj',c}(.)$. Then, the BBAs $m_{jj',p}(.)$ and $m_{jj',c}(.)$ are combined with PCR5 rule to get the BBAs $m_{jj'}(.)$ and, finally, the global Inter-

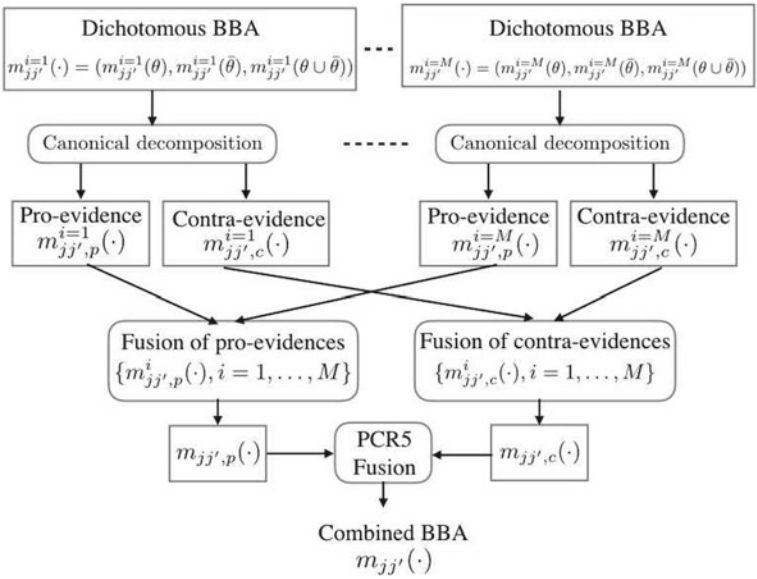


Fig. 1 Principle of fast fusion of $m^i_{jj'}(.)$ of Step 2 of BF-ICrA

Criteria Belief Matrix $\mathbf{K} = [K_{jj'} = (m_{jj'}(\theta), m_{jj'}(\bar{\theta}), m_{jj'}(\theta \cup \bar{\theta}))]$. The principle of this modified step 2 of BF-ICrA is summarized in the Fig. 1 for convenience.

Another simpler fusion method to combine the dichotomous BBAs $m^i_{jj'}(.)$ would just consist to average them. In Sect. 6, we will show how these two methods behave in the examples chosen for the evaluation of MO-ACO Algorithm for optimal WSN deployment.

5 Multi-objective ACO Algorithm

Recently Wireless Sensor Networks (WSNs) have attracted the attention of the research scientists community, conditioned by a set of challenges: theoretical and practical. WSNs consists of distributed sensor nodes and their main purpose is to monitor the real-time environmental status, based on gathering available sensor information, processing and transmitting the collected data to the specified remote base station. It is a promising technology that is used in a coverage of application requiring minimum human contribution, ranging from civil and military to healthcare and environmental monitoring. One of the key mission of WSN is the full surveillance of the monitoring region with a minimal number of sensors and minimized energy consumption of the network. The lifetime of the sensors is strongly related to the amount of the power loaded in the battery, that is why the control of the energy consumption of sensors is an important active research problem. The small energy storage capacity of sensor nodes intrudes the possibility to gather the information directly to the main base. Because of this they transfer their data to the so called High Energy Communication Node (HECN), which is able to collect the information from across the network and to transmit it to the base computer for processing. The sensors transmit their data to the HECN, either directly or via hops, using closest sensors as communication relays. The WSN can have large numbers of nodes and the problem can be very complex.

In order to solve successfully the key mission of WSNs, in [22], we did apply multi-objective Ant Colony Optimization (ACO) to solve this hard, from the computational point of view, telecommunication problem. The number of ants is one of the key algorithm parameters in the ACO and it is important to find the optimal number of ants needed to achieve good solutions with minimal computational resources. In [22], the optimal solution was obtained by applying the classical Atanassov's ICrA method. In the next section we will present the results obtained by the fast BF-ICrA approach and compare their results.

The problem of designing a WSN is multi-objective, with two objective functions: (1) one wants to minimize the energy consumption of the nodes in the network, and (2) one wants to minimize the number of nodes. The full coverage of the network and connectivity are considered as constraints. For solving this problem, we have proposed to use a Multi-Objective Ant Colony Optimization (MO-ACO) algorithm in [22] and we have studied the influence of the number of ants on the algorithm performance and quality of the achieved solutions. The computational resources,

which the algorithm needs, are not negligible. The computational resources depends on the size of the solved problem and on the number of ants. The aim is to find a minimal number of ants which allow the algorithm to find good solution for WSN deployment.

The ACO algorithm uses a colony of artificial ants that behave as cooperating agents. With the help of the pheromone and the heuristic information they try to construct better solutions and to find the optimal ones. The pheromone corresponds to the global memory of the ants and the heuristic information is a some preliminary knowledge of the problem. The problem is represented by a graph and the solution is represented by a path in the graph or by tree in the graph. Ants start from random nodes and construct feasible solutions. When all ants construct their solution the pheromone is updated. The new, added, pheromone depends to the quality of the solution. The elements of the graph, which belong to better solutions will receive more pheromone and will be more desirable in the next iteration. In our implementation, we use the MAX-MIN Ant System (MMAS) which is one of the most successful ant approaches originally presented in [23]. In our case, the graph of the problem is represented by a square grid. The nodes of the graph are enumerated. The ants will deposit their pheromone on the nodes of the grid. We will deposit the sensors on the nodes of the grid too. The solution is represented by tree. An ant starts to create a solution starting from random node, which communicates with the HECN. After it includes next nodes in the solution applying probabilistic rule, called transition probability which is a product of the heuristic information and quantity of the pheromone, corresponding to this new node. Construction of the heuristic information is a crucial point in the ant algorithms. It is problem dependent and helps us to manage the search process. Our heuristic information represented by (21) is a product of three values.

$$\eta_{ij}(t) = s_{ij} l_{ij} (1 - b_{ij}) \tag{21}$$

where s_{ij} is the number of the new points (nodes of the graph) which the new sensor will cover, and which are not covered by other sensors, and

$$l_{ij} = \begin{cases} 1 & \text{if communication exists ;} \\ 0 & \text{if there is no communication.} \end{cases} \tag{22}$$

and where b_{ij} is the solution matrix. The matrix element b_{ij} equals 1 when there is sensor on this position, otherwise $b_{ij} = 0$. With s_{ij}, we try to increase the number of points covered by one sensor and thus to decrease the number of sensors we need. With l_{ij}, we guarantee that all sensors will be connected. With b_{ij} we guarantee that maximum one sensor will be mapped on the same point. The search stops when transition probability $p_{ij} = 0$ for all values of i and j. It means that there are no more free positions, or that all area is fully covered. At the end of every iteration the quantity of the pheromone is updated according to the rule:

$$\tau_{ij} \leftarrow \rho \tau_{ij} + \Delta \tau_{ij}, \tag{23}$$

with the increment $\Delta\tau_{ij} = 1/F(k)$ if (i, j) belongs to the non-dominated solution constructed by ant k, or $\Delta\tau_{ij} = 0$ otherwise. The parameter ρ is a pheromone decreasing parameter chosen in [0, 1]. This parameter ρ models evaporation in the nature and decreases the influence of old information on the search process. After that, we add the new pheromone, which is proportional to the value of the fitness function constructed as:

$$F(k) = \frac{f_1(k)}{\max_i(f_1(i))} + \frac{f_2(k)}{\max_i(f_2(i))}, \tag{24}$$

where $f_1(k)$ is the number of sensors proposed by the kth ant, and $f_2(k)$ is the energy of the solution of the kth ant. These are also the objective functions of the WSN layout problem. We normalize the values of two objective functions with their maximal achieved values from the first iteration.

6 Application to WSN Layout Deployment

In this section we present the results of the fast BF-ICrA method with the MO-ACO algorithm for WSN layout deployment. Fidanova and Roeva have developed a software, which realizes the MO-ACO algorithm. This software can solve the problem at any rectangular area, the communication and the coverage radius can be different and can have any positive value. We can have regions in the area. The program was written in C language, and the tests were run on computer with an Intel Pentium 2.8 GHz processor. In their tests, they use an example where the area is square. The coverage and communication radii cover 30 points. The HECN is fixed in the centre of the area. In the sequel we consider three examples of areas with three sizes: 350 × 350 points, 500 × 500 points, and 700 × 700 points. The MO-ACO algorithm is based on 30 runs for each number of ants. We extract the Pareto front from the solutions of these 30 runs, and we show the achieved non dominated solutions (approximate Pareto fronts) for each case on which the BF-ICrA will be applied. The score matrices for each case is given in Tables 1, 2 and 3 [22].

Table 1 The 6 × 10 score matrix **S** for 350 × 350 case (Example 1)

$$\mathbf{S} = \begin{array}{c} \\ 111 \\ 112 \\ 113 \\ 114 \\ 115 \\ 116 \end{array} \begin{array}{c} \begin{array}{cccccccccc} \mathrm{ACO}_1 & \mathrm{ACO}_2 & \mathrm{ACO}_3 & \mathrm{ACO}_4 & \mathrm{ACO}_5 & \mathrm{ACO}_6 & \mathrm{ACO}_7 & \mathrm{ACO}_8 & \mathrm{ACO}_9 & \mathrm{ACO}_{10} \end{array} \\ \begin{bmatrix} 30 & 36 & 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 36 & 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 28 & 35 & 28 & 30 & 30 & 30 & 28 & 28 & 28 & 28 \\ 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 \\ 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 & 26 \\ 26 & 26 & 26 & 26 & 26 & 26 & 25 & 25 & 26 & 25 \end{bmatrix} \end{array}$$

Table 2 The 22 × 10 score matrix **S** for 500 × 500 case (Example 2)

	ACO_1	ACO_2	ACO_3	ACO_4	ACO_5	ACO_6	ACO_7	ACO_8	ACO_9	ACO_{10}
223	90	96	90	90	89	81	90	90	90	90
224	61	96	89	89	88	65	61	59	57	71
225	61	96	74	58	60	58	57	58	57	57
226	59	95	73	57	59	57	56	58	57	57
227	60	57	57	57	57	56	56	57	57	57
228	60	57	57	57	57	56	56	57	54	57
229	58	57	57	55	57	56	56	56	54	56
230	57	57	57	55	57	52	56	54	54	56
231	57	55	57	55	55	52	56	54	54	56
232	57	55	55	51	54	50	52	51	54	48
233	57	55	55	51	54	50	51	51	54	48
234	57	55	55	51	53	50	51	48	53	48
235	57	55	54	51	53	50	51	48	50	48
236	57	55	54	51	53	50	51	48	50	48
237	57	55	54	51	53	50	51	48	50	48
238	57	55	53	51	53	50	51	48	50	48
239	56	55	53	50	53	50	51	48	50	48
240	53	53	53	50	53	50	51	48	50	48
241	53	53	53	50	53	50	51	48	50	48
242	53	53	53	50	53	50	51	48	50	48
243	53	53	53	50	53	50	51	48	50	48
244	53	53	53	50	52	50	51	48	50	48

Table 3 The 19 × 10 score matrix **S** for 700 × 700 case (Example 3)

	ACO_1	ACO_2	ACO_3	ACO_4	ACO_5	ACO_6	ACO_7	ACO_8	ACO_9	ACO_{10}
437	173	173	173	173	173	118	168	172	261	172
438	173	173	173	173	173	118	112	117	260	172
439	172	173	173	173	140	93	110	115	131	172
440	172	173	173	173	115	93	110	114	111	162
441	172	173	173	122	111	93	110	114	111	110
442	172	173	173	114	111	93	110	112	111	110
443	172	150	123	114	111	93	110	112	111	110
444	124	112	112	106	107	93	110	102	111	105
445	117	112	112	106	107	93	110	102	108	105
446	117	112	105	105	105	93	107	102	104	105
447	117	112	105	105	105	93	105	102	102	105
448	115	111	105	105	105	93	105	102	102	105
449	115	111	105	105	105	93	102	99	102	105
450	113	111	105	105	105	93	102	99	102	105
451	113	109	105	105	105	93	102	99	97	105
452	113	109	105	105	105	93	99	99	97	104
453	113	109	105	105	105	93	99	99	97	104
454	113	109	105	105	96	93	96	96	96	104
455	106	106	105	105	96	93	96	96	96	97

Table 4 Matrix $\mathbf{K}_{\approx PCR6}(\theta)$ for Example 1

$$\begin{bmatrix} 0.865 & 0.821 & 0.865 & 0.790 & 0.790 & 0.790 & 0.806 & 0.806 & 0.865 & 0.806 \\ 0.821 & 0.928 & 0.821 & 0.950 & 0.950 & 0.950 & 0.805 & 0.805 & 0.821 & 0.805 \\ 0.865 & 0.821 & 0.865 & 0.790 & 0.790 & 0.790 & 0.806 & 0.806 & 0.865 & 0.806 \\ 0.790 & 0.950 & 0.790 & 1.000 & 1.000 & 1.000 & 0.795 & 0.795 & 0.790 & 0.795 \\ 0.790 & 0.950 & 0.790 & 1.000 & 1.000 & 1.000 & 0.795 & 0.795 & 0.790 & 0.795 \\ 0.790 & 0.950 & 0.790 & 1.000 & 1.000 & 1.000 & 0.795 & 0.795 & 0.790 & 0.795 \\ 0.806 & 0.805 & 0.806 & 0.795 & 0.795 & 0.795 & 0.843 & 0.843 & 0.806 & 0.843 \\ 0.806 & 0.805 & 0.806 & 0.795 & 0.795 & 0.795 & 0.843 & 0.843 & 0.806 & 0.843 \\ 0.865 & 0.821 & 0.865 & 0.790 & 0.790 & 0.790 & 0.806 & 0.806 & 0.865 & 0.806 \\ 0.806 & 0.805 & 0.806 & 0.795 & 0.795 & 0.795 & 0.843 & 0.843 & 0.806 & 0.843 \end{bmatrix}$$

Each row of **S** corresponds to the number of sensors used in WSN to cover the area as indicated in the first column at the left side of the score matrix. Each column of **S** corresponds to ACO_j algorithm used with j ants ($j = 1, 2, \ldots, 10$). Each element S_{ij} of **S** corresponds to the energy corresponding to this number of sensors and with the number of ants used for Multiple Objective ACO algorithm.

6.1 Application of Fast BF-ICrA in Example 1 (350 × 350 Points)

In this example, one sees from the score matrix of the Table 1 that ACO_1, ACO_3 and ACO_9 algorithms perform equally for all alternatives (i.e. all rows) and they define a first group/cluster of methods providing exactly the same performances. Similarly, ACO_4, ACO_5 and ACO_6 constitute a second group of algorithms. The third group is made of ACO_7, ACO_8 and ACO_{10} algorithms. It is worth noting that these three groups $\{\mathrm{ACO}_1, \mathrm{ACO}_3, \mathrm{ACO}_9\}$, $\{\mathrm{ACO}_4, \mathrm{ACO}_5, \mathrm{ACO}_6\}$, and $\{\mathrm{ACO}_7, \mathrm{ACO}_8, \mathrm{ACO}_{10}\}$ differ only very slightly, whereas the ACO_2 algorithm (i.e the 2nd column of the score matrix **S**) differs a bit more from all the three aforementioned groups.

Example 1 with fast PCR6: If we apply the fast BF-ICrA method using approximate PCR6 fusion rule based on the canonical decomposition of the $M = 6$ dichotomous BBAs $(m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}))$, we get the matrix of mass of belief of agreement between criteria given in Table 4.[6]

The matrix of distances to full agreement based on fast BF-ICrA method, denoted by $\mathbf{D}_{\approx PCR6}(\theta)$, is given in Table 5.

In examining the Table 5, one sees that $ACO1$, $ACO3$ and $ACO9$ are at a small distance 0.134, with respect to other algorithms, so that they belong to the same

[6] All the numerical values presented in the matrices have been truncated at their 3rd digit for typesetting convenience.

Table 5 Matrix $\mathbf{D}_{\approx PCR6}(\theta)$ with fast BF-ICrA for Example 1

$$\begin{bmatrix} 0.134 & 0.178 & \mathbf{0.134} & 0.209 & 0.209 & 0.209 & 0.193 & 0.193 & \mathbf{0.134} & 0.193 \\ 0.178 & 0.071 & 0.178 & 0.049 & 0.049 & 0.049 & 0.194 & 0.194 & 0.178 & 0.194 \\ \mathbf{0.134} & . 0.178 & 0.134 & 0.209 & 0.209 & 0.209 & 0.193 & 0.193 & 0.134 & 0.193 \\ 0.209 & 0.049 & 0.209 & 0 & \mathbf{0} & \mathbf{0} & 0.204 & 0.204 & 0.209 & 0.204 \\ 0.209 & 0.049 & 0.209 & \mathbf{0} & 0 & \mathbf{0} & 0.204 & 0.204 & 0.209 & 0.204 \\ 0.209 & 0.049 & 0.209 & \mathbf{0} & \mathbf{0} & 0 & 0.204 & 0.204 & 0.209 & 0.204 \\ 0.193 & 0.194 & 0.193 & 0.204 & 0.204 & 0.204 & 0.156 & \mathbf{0.156} & 0.193 & \mathbf{0.156} \\ 0.193 & 0.194 & 0.193 & 0.204 & 0.204 & 0.204 & \mathbf{0.156} & 0.156 & 0.193 & \mathbf{0.156} \\ \mathbf{0.134} & 0.178 & 0.134 & 0.209 & 0.209 & 0.209 & 0.193 & 0.193 & 0.134 & 0.193 \\ 0.193 & 0.194 & 0.193 & 0.204 & 0.204 & 0.204 & \mathbf{0.156} & \mathbf{0.156} & 0.193 & 0.156 \end{bmatrix}$$

Table 6 Matrix $\mathbf{D}_{\text{Aver.}}(\theta)$ with BF-ICrA using averaging rule for Example 1

$$\begin{bmatrix} 0.084 & 0.082 & 0.084 & 0.081 & 0.081 & 0.081 & 0.156 & 0.156 & 0.084 & 0.156 \\ 0.082 & 0.030 & 0.082 & 0.016 & 0.016 & 0.016 & 0.142 & 0.142 & 0.082 & 0.142 \\ 0.084 & 0.082 & 0.084 & 0.081 & 0.081 & 0.081 & 0.156 & 0.156 & 0.084 & 0.156 \\ 0.081 & 0.016 & 0.081 & 0 & 0 & 0 & 0.138 & 0.138 & 0.081 & 0.138 \\ 0.081 & 0.016 & 0.081 & 0 & 0 & 0 & 0.138 & 0.138 & 0.081 & 0.138 \\ 0.081 & 0.016 & 0.081 & 0 & 0 & 0 & 0.138 & 0.138 & 0.081 & 0.138 \\ 0.156 & 0.142 & 0.156 & 0.138 & 0.138 & 0.138 & 0.198 & 0.198 & 0.156 & 0.198 \\ 0.156 & 0.142 & 0.156 & 0.138 & 0.138 & 0.138 & 0.198 & 0.198 & 0.156 & 0.198 \\ 0.084 & 0.082 & 0.084 & 0.081 & 0.081 & 0.081 & 0.156 & 0.156 & 0.084 & 0.156 \\ 0.156 & 0.142 & 0.156 & 0.138 & 0.138 & 0.138 & 0.198 & 0.198 & 0.156 & 0.198 \end{bmatrix}$$

group and behave similarly. Same remarks holds for the group $\{ACO_4, ACO_5, ACO_6\}$ because its inter-distance is zero, and for the group $\{ACO_7, ACO_8, ACO_{10}\}$ because its inter-distance is 0.156. In a relative manner ACO_2 appears closer to $\{ACO_4, ACO_5, ACO_6\}$, than $\{ACO_1, ACO_3, ACO_9\}$ or $\{ACO_7, ACO_8, ACO_{10}\}$, which intuitively makes sense when comparing directly the columns of the matrix of Table 1.

Example 1 with averaging fusion: The matrix of distances to full agreement based on BF-ICrA method using average fusion rule, denoted by $\mathbf{D}_{\text{Aver.}}(\theta)$, is given in Table 6.

One sees that only the group $\{ACO_4, ACO_5, ACO_6\}$ can be clearly identified based on the averaging fusion rule. The other groups ACO_2 appears also close to $\{ACO_4, ACO_5, ACO_6\}$. But ACO_1, ACO_3 and ACO_9 are closer to $\{ACO_4, ACO_5, ACO_6\}$ also than in-between. Same remarks holds for ACO_7, ACO_8, and ACO_{10}. So one sees that the averaging fusion rule is not recommended for making the BF-ICrA in this example.

6.2 Application of Fast BF-ICrA in Example 2 (500 × 500 Points)

Example 2 with fast PCR6: If we apply the fast BF-ICrA method using approximate PCR6 fusion rule based on the canonical decomposition of the $M = 22$ dichotomous BBAs $(m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}))$, we get the following matrix of distances to full agreement, denoted by $\mathbf{D}_{\approx PCR6}(\theta)$, given in Table 7.

Based on these results, one sees that no clear group can be identified but we emphasize in boldface in Table 7 the minimal value for each row of the distance matrix $\mathbf{D}_{\approx PCR6}(\theta)$ (diagonal elements excluded). We see that ACO_2 is at the farthest distance of ACO_1 because $D_{12}(\theta) = 0.376$, but in the mean time ACO_2 is at closest distance to ACO_1 because $D_{2j}(\theta) > 0.376$ (for $j > 2$) as shown in second line of Table 7. So we can conclude that ACO_2 is not close to any other algorithm in fact. If we choose a ad-hoc distance threshold, say for instance 0.28, then we can identify the group $\{ACO_1, ACO_7, ACO_8, ACO_9\}$.

Example 2 with averaging fusion: The matrix of distances to full agreement based on BF-ICrA method using average fusion rule, denoted by $\mathbf{D}_{\text{Aver.}}(\theta)$, is given in Table 8.

Based on the average fusion rule there is no clear clustering of algorithms. However based on shortest inter-distance we could make the following distinct pairwise groupings $\{ACO_2, ACO_3\}$, $\{ACO_6, ACO_7\}$, $\{ACO_4, ACO_{10}\}$, $\{ACO_8, ACO_9\}$ and $\{ACO_1, ACO_5\}$ if necessary, but remember that average fusion rule cannot provide the best result as shown in Example 1.

Table 7 Matrix $\mathbf{D}_{\approx PCR6}(\theta)$ with fast BF-ICrA for Example 2

$$\begin{bmatrix}
0.158 & 0.376 & 0.338 & 0.300 & 0.286 & 0.279 & 0.247 & 0.251 & \mathbf{0.225} & 0.280 \\
\mathbf{0.376} & 0.324 & 0.426 & 0.456 & 0.437 & 0.453 & 0.457 & 0.433 & 0.435 & 0.449 \\
\mathbf{0.338} & 0.426 & 0.407 & 0.411 & 0.382 & 0.423 & 0.418 & 0.402 & 0.393 & 0.414 \\
\mathbf{0.300} & 0.456 & 0.411 & 0.349 & 0.323 & 0.381 & 0.368 & 0.370 & 0.362 & 0.363 \\
\mathbf{0.286} & 0.437 & 0.382 & 0.323 & 0.284 & 0.348 & 0.334 & 0.334 & 0.328 & 0.333 \\
\mathbf{0.279} & 0.453 & 0.423 & 0.381 & 0.348 & 0.316 & 0.298 & 0.317 & 0.308 & 0.308 \\
\mathbf{0.247} & 0.457 & 0.418 & 0.368 & 0.334 & 0.298 & 0.235 & 0.276 & 0.255 & 0.283 \\
\mathbf{0.251} & 0.433 & 0.402 & 0.370 & 0.334 & 0.317 & 0.276 & 0.265 & 0.260 & 0.303 \\
\mathbf{0.225} & 0.435 & 0.393 & 0.362 & 0.328 & 0.308 & 0.255 & 0.260 & 0.211 & 0.304 \\
\mathbf{0.280} & 0.449 & 0.414 & 0.363 & 0.333 & 0.308 & 0.283 & 0.303 & 0.304 & 0.277
\end{bmatrix}$$

Table 8 Matrix $\mathbf{D}_{\text{Aver.}}(\theta)$ with BF-ICrA using averaging rule for Example 2

$$\begin{bmatrix} 0.361 & 0.316 & 0.310 & 0.311 & 0.336 & 0.300 & 0.306 & 0.316 & 0.320 & 0.309 \\ 0.316 & 0.125 & 0.158 & 0.198 & 0.225 & 0.187 & 0.216 & 0.225 & 0.240 & 0.206 \\ 0.310 & 0.158 & 0.165 & 0.185 & 0.215 & 0.178 & 0.200 & 0.215 & 0.227 & 0.193 \\ 0.311 & 0.198 & 0.185 & 0.183 & 0.216 & 0.181 & 0.197 & 0.217 & 0.231 & 0.192 \\ 0.336 & 0.225 & 0.215 & 0.216 & 0.243 & 0.214 & 0.231 & 0.249 & 0.261 & 0.226 \\ 0.300 & 0.187 & 0.178 & 0.181 & 0.214 & 0.159 & 0.175 & 0.194 & 0.210 & 0.176 \\ 0.306 & 0.216 & 0.200 & 0.197 & 0.231 & 0.175 & 0.181 & 0.202 & 0.216 & 0.186 \\ 0.316 & 0.225 & 0.215 & 0.217 & 0.249 & 0.194 & 0.202 & 0.215 & 0.229 & 0.204 \\ 0.320 & 0.240 & 0.227 & 0.231 & 0.261 & 0.210 & 0.216 & 0.229 & 0.233 & 0.222 \\ 0.309 & 0.206 & 0.193 & 0.192 & 0.226 & 0.176 & 0.186 & 0.204 & 0.222 & 0.183 \end{bmatrix}$$

Table 9 Matrix $\mathbf{D}_{\approx PCR6}(\theta)$ with fast BF-ICrA for Example 3

$$\begin{bmatrix} 0.313 & 0.388 & 0.465 & 0.498 & 0.469 & 0.500 & 0.426 & 0.451 & 0.498 & 0.477 \\ 0.388 & 0.339 & 0.403 & 0.496 & 0.461 & 0.500 & 0.421 & 0.440 & 0.497 & 0.464 \\ 0.465 & 0.403 & 0.348 & 0.493 & 0.456 & 0.500 & 0.416 & 0.437 & 0.495 & 0.457 \\ 0.498 & 0.496 & 0.493 & 0.362 & 0.385 & 0.500 & 0.376 & 0.391 & 0.470 & 0.303 \\ 0.469 & 0.461 & 0.456 & 0.385 & 0.230 & 0.380 & 0.256 & 0.288 & 0.300 & 0.324 \\ 0.500 & 0.500 & 0.500 & 0.500 & 0.380 & 0 & 0.312 & 0.356 & 0.308 & 0.500 \\ 0.426 & 0.421 & 0.416 & 0.376 & 0.256 & 0.312 & 0.137 & 0.185 & 0.272 & 0.330 \\ 0.451 & 0.440 & 0.437 & 0.391 & 0.288 & 0.356 & 0.185 & 0.205 & 0.314 & 0.351 \\ 0.498 & 0.497 & 0.495 & 0.470 & 0.300 & 0.308 & 0.272 & 0.314 & 0.283 & 0.438 \\ 0.477 & 0.464 & 0.457 & 0.303 & 0.324 & 0.500 & 0.330 & 0.351 & 0.438 & 0.228 \end{bmatrix}$$

6.3 Application of Fast BF-ICrA in Example 3 (700 × 700 Points)

Example 3 with fast PCR6: If we apply the fast BF-ICrA method using approximate PCR6 fusion rule based on the canonical decomposition of the $M = 19$ dichotomous BBAs $(m^i_{jj'}(\theta), m^i_{jj'}(\bar{\theta}), m^i_{jj'}(\theta \cup \bar{\theta}))$, we get the matrix of distances to full agreement, denoted by $\mathbf{D}_{\approx PCR6}(\theta)$, given in Table 9.

We observe that the average distance between ACO algorithms is much higher than in Tables 5 and 7 of Examples 1 and 2. This shows clearly the difficulty to precisely identify the clusters of similar algorithms because only few ACO algorithms perform actually very well for this third example. Eventually, and based on shortest inter-distance we could make the first pairwise group $\{ACO_7, ACO_8\}$ because $D_{78}(\theta) = 0.185$ is the minimal inter-distance we have between the ACO algorithms. Once the rows and columns of Table 9 corresponding to ACO_7 and ACO_8 are eliminated, then the second best group will be $\{ACO_5, ACO_9\}$ because $D_{59}(\theta) = 0.300$. Similarly, we will get the group $\{ACO_4, ACO_{10}\}$ because $D_{4,10}(\theta) = 0.303$, and then the group

Table 10 Matrix $\mathbf{D}_{\text{Aver.}}(\theta)$ with BF-ICrA using averaging rule for Example 3

$$\begin{bmatrix} 0.170 & 0.154 & 0.142 & 0.221 & 0.351 & 0.350 & 0.392 & 0.345 & 0.332 & 0.298 \\ 0.154 & 0.120 & 0.092 & 0.167 & 0.321 & 0.295 & 0.369 & 0.313 & 0.290 & 0.261 \\ 0.142 & 0.092 & 0.042 & 0.114 & 0.289 & 0.237 & 0.342 & 0.279 & 0.242 & 0.224 \\ 0.221 & 0.167 & 0.114 & 0.054 & 0.255 & 0.139 & 0.327 & 0.260 & 0.184 & 0.177 \\ 0.351 & 0.321 & 0.289 & 0.255 & 0.339 & 0.245 & 0.391 & 0.355 & 0.287 & 0.324 \\ 0.350 & 0.295 & 0.237 & 0.139 & 0.245 & 0 & 0.304 & 0.242 & 0.115 & 0.247 \\ 0.392 & 0.369 & 0.342 & 0.327 & 0.391 & 0.304 & 0.390 & 0.368 & 0.336 & 0.387 \\ 0.345 & 0.313 & 0.279 & 0.260 & 0.355 & 0.242 & 0.368 & 0.328 & 0.288 & 0.341 \\ 0.332 & 0.290 & 0.242 & 0.184 & 0.287 & 0.115 & 0.336 & 0.288 & 0.190 & 0.279 \\ 0.298 & 0.261 & 0.224 & 0.177 & 0.324 & 0.247 & 0.387 & 0.341 & 0.279 & 0.261 \end{bmatrix}$$

$\{ACO_1, ACO_2\}$ because $D_{12}(\theta) = 0.388$. Finally we could also cluster ACO_3 with ACO_6 because $D_{36}(\theta) = 0.500$, although this distance of agreement is quite large to be considered as a trustable cluster.

Example 3 with averaging fusion: The matrix of distances to full agreement based on BF-ICrA method using average fusion rule, denoted by $\mathbf{D}_{\text{Aver.}}(\theta)$, is given in Table 10.

Surprisingly, the use of averaging rule provides in this example lower distance values on average with respect to values given in Table 9. However no clear clustering of algorithms can be made because only few ACO algorithms perform actually very well for this third example. If we adopt the pairwise strategy to cluster algorithms, we will obtain now as first group $\{ACO_2, ACO_3\}$ because $D_{23}(\theta) = 0.092$, as second group $\{ACO_6, ACO_9\}$ because $D_{69}(\theta) = 0.115$, as third group $\{ACO_4, ACO_{10}\}$ because $D_{4,10}(\theta) = 0.177$, as fourth group $\{ACO_1, ACO_8\}$ because $D_{18}(\theta) = 0.345$, and finally we could also cluster ACO_5 with ACO_7 because $D_{57}(\theta) = 0.391$. One sees that there is no strong correlation between results obtained from BF-ICrA based on fast PCR6 and those based on averaging rule, which is not surprising because the rules are totally different. Nevertheless the group $\{ACO_4, ACO_{10}\}$ is agreed by both methods here.

7 Application to Workforce Planning Problem (WPP)

In this section we present a new application of our new Fast BF-ICrA method for solving the workforce planning problem (WPP). This problem has been addressed recently by Fidanova et al. in [24] using the classical Atanassov's ICrA method [2–5]. Before presenting our new results, it is necessary to present briefly the WPP.

7.1 The Workforce Planning Problem (WPP)

The workforce planning is a part of the human resource management. It includes multiple level of complexity, therefore it is a hard optimization problem (NP-hard). This problem consists of two decision sets: selection and assignment. The first set shows selected employees from available workers. The assignment set shows which worker which job will perform. The aim is to fulfill the work requirements with minimal assignment cost. Such hard optimization problem with strong constraints is usually impossible to solve with exact methods or traditional numerical methods for instances with realistic size and that is why these methods (exact or numerical) can be applied only on some simplified variants of the original problem (see [24] for a detailed bibliography of the existing methods). One must emphasize that the convex optimization methods are not applicable complex non-linear workforce planning problems. Nowadays, nature-inspired metaheuristic methods receive great attention [25–29]. In the WPP considered heresome heuristic method including genetic algorithm [31, 32], memetic algorithm [30], scatter search [31] etc., have been already applied. So far the Ant Colony Optimization (ACO) algorithm is proved to be very effective solving various complex optimization problems [33, 34]. In our previous work [35] we did propose ACO algorithm for workforce planning. We have considered the variant of the workforce planning problem proposed in [31]. More recently in [24] we proposed a hybrid ACO algorithm which is a combination of ACO with a local search procedure and the classical InterCriteria Analysis (ICrA) was used to analyze the algorithm performance according the local search procedures and to study the correlations between the different variants in order to improve the algorithm performance for solving efficiently the WPP.

In this section we solve the WPP proposed in [31, 36]. The set of jobs $J = \{1, \ldots, m\}$ must be completed during a fixed period of time. The job j requires d_j hours to be completed. $I = \{1, \ldots, n\}$ is the set of workers, candidates to be assigned. The minimal number of hours that every job must require by every assigned worker is h_{min}. Availability of the worker i is s_i hours. One worker can be assigned to maximum j_{max} jobs. The set A_i shows the jobs, that worker i is qualified. Maximum t workers can be assigned during the planed period, or at most t workers may be selected from the set I of workers. The selected workers need to be capable to complete all the jobs. The aim is to find feasible solution, that optimizes a given objective function.

Let c_{ij} is the cost of assigning the worker i to the job j. The mathematical model of the WPP can be described by the following variables

$$x_{ij} \triangleq \begin{cases} 1, & \text{if worker } i \text{ is assigned to job } j, \\ 0, & \text{otherwise.} \end{cases} \quad \text{and } y_i \triangleq \begin{cases} 1, & \text{if worker } i \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

$$z_{ij} \triangleq \text{number of hours that worker } i \text{ is assigned to perform job } j$$

$$Q_j \triangleq \text{set of workers qualified to perform job } j$$

The WPP consists to minimize the total assignment cost under some constraints, more precisely we want to

$$\text{Minimize} \sum_{i \in I} \sum_{j \in A_i} c_{ij}.x_{ij} \tag{25}$$

subject to the following constraints

$$\sum_{j \in A_i} z_{ij} \leq s_i.y_i, \qquad \text{with } i \in I \tag{26}$$

$$\sum_{i \in Q_j} z_{ij} \geq d_j \qquad \text{with } j \in J \tag{27}$$

$$\sum_{j \in A_i} x_{ij} \leq j_{max}.y_j \qquad \text{with } i \in I \tag{28}$$

$$h_{min}.x_{ij} \leq z_{ij} \leq s_i.x_{ij} \qquad \text{with } i \in I,\ j \in A_i \tag{29}$$

$$\sum_{i \in I} y_i \leq t \tag{30}$$

where $x_{ij} \in \{0, 1\}$, $y_i \in \{0, 1\}$ and $z_{ij} \geq 0$ with $i \in I$ and $j \in A_i$.

The constraint (26) stipulates that number of hours for each selected worker is limited, and the constraint (27) stipulates that the work must be totally achieved. The number of jobs that every worker can perform is limited according to the constraint (28). The inequality (29) stipulates the minimal number of hours that every job must require by every assigned worker. The number of assigned workers is limited by the constraint (30). WPP is difficult to solve because of very restrictive constraints especially the relation between the parameters h_{min} and d_j. It is easier to solve (to find feasible solution) when the problem is structured (when d_j is a multiple of h_{min}) than for unstructured problems (when d_j and h_{min} are not related).

Based on our previous works [24, 35], we will apply the ACO algorithm for workforce planning problem coupled with different local search procedures and based on our new fast BF-ICrA approach. One of the main points of the ant algorithm is the proper representation of the problem by graph. In our case the graph of the problem is 3 dimensional and the node (i, j, z) corresponds worker with number i to be assigned to the job j for time z. The graph of the problem is asymmetric, because the maximal value of z depends of the value of j, different jobs needs different time to be completed. At the beginning of every iteration every ant starts to construct their solution, from random node of the graph of the problem. For every ant, three random numbers are generated. The first random number corresponds to a worker we randomly select in the interval $[0, \ldots, n]$. The second random number in the interval $[0, \ldots, m]$ corresponds to the job that must be done by the worker. We check if the worker is qualified to perform the job, if not we randomly choose for him/her another compatible job. The third random number in $[h_{min}, \ldots, \min\{d_j, s_i\}]$ corresponds to

the number of hours allocated to worker i to performs the job j. After, the ant applies the transition probability rule to include next nodes in the partial solution, until a feasible solution is completed, or there is no possibility to include new node. The heuristic information η_{ijl} is problem dependent and is used for better management of the search process. In order to assign the most cheapest worker as longer as possible, we define the η_{ijl} parameter of the ACO algorithm by:

$$\eta_{ijl} = \begin{cases} l/c_{ij} & \text{if } l = z_{ij} \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

The node with a highest probability is chosen to be the next node, included in the solution. When there are several candidate nodes with a same probability, the next node is randomly drawn among these candidates. When some move of the ant do not meets the problem constraints, then the probability of this move is set to be 0. If for all possible nodes the value of the transition probability is 0, it is impossible to include new node in the solution and the solution construction stops. When the constructed solution is feasible, the value of the objective function is the sum of the assignment cost of the assigned workers. If the constructed solution is not feasible, the value of the objective function is set to be equal to -1. The ants construct feasible solutions and depose a new pheromone on the elements of their solutions. More precisely, the main pheromone trail update rule is given by

$$\tau_{i,j} \leftarrow \rho\tau_{i,j} + \Delta\tau_{i,j}, \tag{32}$$

where ρ decreases the value of the pheromone (like the evaporation in a nature), and where the new added pheromone $\Delta\tau_{i,j}$ is equal to the reciprocal value of the objective function given by

$$\Delta\tau_{i,j} = \frac{\rho - 1}{\min \sum_{i \in I} \sum_{j \in A_i} c_{ij} . x_{ij}} \tag{33}$$

The nodes of the graph belonging to solutions with less value of the objective function, receive more pheromone than others and become more desirable in the next iteration. At the end of every iteration we compare the iteration best solution with the best solution obtained so far. If the best solution from the current iteration is better than the best so far solution (global best solution), we update the global best solution with the current iteration best solution. The end condition used in our algorithm is the number of iterations.

In order to decrease the time to find the best solution and eventually to improve the achieved solutions, we use the local search proposed in [24] because it increases the possibility to find feasible solution and thus the chance to improve current solution. If the solution is not feasible we remove part of the assigned workers and after that we assign in their place new workers. The workers which will be removed are chosen randomly. On this partial solution we assign new workers applying the rules of ant

algorithm. The ACO algorithm (denoted ACO_1) is a stochastic algorithm, therefore the new constructed solution is different from previous one with a high probability. We have proposed three variants for the local search procedure:

- ACO_2: with removed workers are quarter of all assigned workers (ACO quarter);
- ACO_3: with removed workers are half of all assigned workers; (ACO half)
- ACO_4: with all assigned workers are removed and the solution is constructed from the beginning (ACO restart).

7.2 WPP Addressed in This Paper

We use the artificially generated problem instances considered in [31]. The characteristics of this WPP are: $m = 20$, $n = 20$, $t = 10$, $s_i \in [50, 70]$, $j_{\max} \in [3, 5]$, and $h_{\min} \in [10, 15]$. The set of test problems consists of ten structured problems, and ten unstructured problems. For structured problems d_j is proportional to $h_{\min}$. In our previous work [35] we have shown that our ACO algorithm outperforms the genetic and scatter search algorithms presented in [31]. The number of iterations is a stopping criteria for our hybrid ACO algorithm. The ACO parameter settings are as follows: $\rho = 0.5, \tau_0 = 0.5, a = 1, b = 1$, the number of ants is 20, and the maximum number of iterations is fixed to 100. Further, the problem instances are enumerated as $S20_{01}$ to $S20_{10}$ for the ten structured problems using 20 ants, and as $U20_{01}$ to $U20_{10}$ for the ten unstructured problems using 20 ants. The WPP has very restrictive constraints. Therefore only 2–3 of the ants, per iteration, find feasible solution. Sometimes some iterations do not generate a feasible solution. Its complicates the search process. Our aim is to decrease the number of unfeasible solutions, in order to increase the possibility for ants to find good solutions, and therefore to decrease the needed number of iterations to get a good solution. We observe that after the local search procedure applied on the first iteration, the number of unfeasible solutions in a next iterations decreases. It is another reason why the computation time does not increase significantly. We are analyzing four cases: (1) without local search procedure (ACO_1), and with the three aforementioned variant for the local search procedure (ACO_2, ACO_3 and ACO_4). We did perform 30 independent runs for each of the four cases (because the algorithm is stochastic) to guarantee the robustness of the average results. We apply ANOVA test for statistical analysis to guarantee the significance of the achieved results. The obtained results are presented in Tables 11 and 12. Tables 11 presents the minimal number of iterations to achieve the best solution and Table 12—the computation time needed to achieve the best solution.

Table 11 Minimal number of iterations to achieve the best solution

Algo type	ACO_1	ACO_2	ACO_3	ACO_4
$S20_{01}$	13	10	15	16
$S20_{02}$	17	28	28	35
$S20_{03}$	29	27	37	33
$S20_{04}$	77	66	41	23
$S20_{05}$	21	21	4	14
$S20_{06}$	21	13	20	1
$S20_{07}$	43	34	29	40
$S20_{08}$	57	15	50	33
$S20_{09}$	36	28	22	48
$S20_{10}$	26	19	16	35
$U20_{01}$	17	23	11	21
$U20_{02}$	17	16	12	15
$U20_{03}$	28	22	20	48
$U20_{04}$	41	56	28	28
$U20_{05}$	14	20	15	4
$U20_{06}$	46	46	45	20
$U20_{07}$	29	44	37	39
$U20_{08}$	11	14	16	26
$U20_{09}$	46	68	41	42
$U20_{10}$	30	30	30	30

7.3 Results of WPP Obtained with Fast BF-ICrA

The test problems $S20_{01}$ to $S20_{10}$ and $U20_{01}$ to $U20_{10}$ are considered as objects, and algorithms ACO_1, ACO_2, ACO_3 and ACO_4 as criteria. We did apply the fast BF-ICrA approach to identify the relation between the proposed ACO hybrid algorithms. The hybrid algorithms are compared based on the obtained results according to the number of iterations (Table 11), and according to the computation time (Table 12).

Based on the values of Table 11, we get the distances between ACO algorithms reported in matrix $\mathbf{D}^{it}_{\approx PCR6}(\theta)$ with fast PCR6 rule, and the matrix $\mathbf{D}^{it}_{Aver.}(\theta)$ when using the simple averaging rule of combination.

$$\mathbf{D}^{it}_{\approx PCR6}(\theta) = \begin{bmatrix} 0.3181 & 0.4419 & 0.4218 & 0.4867 \\ 0.4419 & 0.3743 & 0.4812 & 0.4925 \\ 0.4218 & 0.4812 & 0.3316 & 0.4802 \\ 0.4867 & 0.4925 & 0.4802 & 0.3575 \end{bmatrix}$$

$$\mathbf{D}^{it}_{Aver.}(\theta) = \begin{bmatrix} 0.3736 & 0.4018 & 0.4253 & 0.4885 \\ 0.4018 & 0.3450 & 0.4372 & 0.4834 \\ 0.4253 & 0.4372 & 0.3933 & 0.4791 \\ 0.4885 & 0.4834 & 0.4791 & 0.3792 \end{bmatrix}$$

Table 12 Computation time needed to achieve the best solution

Algo type	ACO_1	ACO_2	ACO_3	ACO_4
$S20_{01}$	1.20	0.94	0.96	2.29
$S20_{02}$	3.94	8.62	6.22	14.75
$S20_{03}$	5.19	5.79	11.93	3.06
$S20_{04}$	3.06	16.66	7.00	6.11
$S20_{05}$	0.63	1.312	0.396	0.90
$S20_{06}$	2.48	2.12	2.64	0.59
$S20_{07}$	6.78	4.82	6.78	6.60
$S20_{08}$	6.38	1.87	10.42	8.59
$S20_{09}$	4.68	5.31	4.48	5.70
$S20_{10}$	1.45	1.25	1.28	10.43
$U20_{01}$	3.10	4.48	2.00	2.50
$U20_{02}$	1.98	1.18	0.92	0.93
$U20_{03}$	2.14	2.41	1.54	1.88
$U20_{04}$	3.08	3.35	3.12	3.47
$U20_{05}$	1.55	2.76	2.06	1.056
$U20_{06}$	10.92	11.8	4.36	7.05
$U20_{07}$	4.22	6.55	3.54	3.27
$U20_{08}$	0.89	1.48	1.19	1.77
$U20_{09}$	6.48	8.72	7.10	7.21
$U20_{10}$	3.74	3.88	3.69	10.00

The analysis of values of $\mathbf{D}^{it}_{\approx PCR6}(\theta)$ matrix shows clearly that none of these algorithms are close of each others because their distances are much bigger than zero. This is because the BBAs of the Inter-Criteria matrix $\mathbf{K}$ are in fact quite ambiguous (i.e. the focal elements θ and $\bar{\theta}$ have comparable mass values), even if there is only a little mass committed to uncertainty $\theta \cup \bar{\theta}$. However, ACO_4 is more distant of other algorithms which indicates a different behavior compared to ACO_1, ACO_2 and ACO_3, as already mentioned in [24]. The averaging fusion rule makes the distances values more close which makes the separability of criteria even more difficult to identify.

Based on the numerical values of Table 12, we get the distances between ACO algorithms reported in matrix $\mathbf{D}^{sec}_{\approx PCR6}(\theta)$ with fast PCR6 rule, and the matrix $\mathbf{D}^{sec}_{Aver.}(\theta)$ when using the simple averaging rule of combination.

$$\mathbf{D}^{sec}_{\approx PCR6}(\theta) = \begin{bmatrix} 0.3021 & 0.4452 & 0.4186 & 0.4603 \\ 0.4452 & 0.3509 & 0.4747 & 0.4807 \\ 0.4186 & 0.4747 & 0.3600 & 0.4798 \\ 0.4603 & 0.4807 & 0.4798 & 0.3595 \end{bmatrix}$$

$$\mathbf{D}^{sec}_{Aver.}(\theta) = \begin{bmatrix} 0.3841 & 0.4030 & 0.4034 & 0.4276 \\ 0.4030 & 0.3215 & 0.3976 & 0.4108 \\ 0.4034 & 0.3976 & 0.3475 & 0.4156 \\ 0.4276 & 0.4108 & 0.4156 & 0.3378 \end{bmatrix}$$

Based on the values of $\mathbf{D}^{sec}_{\approx PCR6}(\theta)$ matrix, one sees also that there is no clear clustering of the different ACO algorithms because the distances values are much bigger than zero, however one can also reasonably infer that ACO_4 shows a different behavior compared to ACO_1, ACO_2 and ACO_3, as inferred in the previous analysis based on input values of Table 11. The difference comes from deleting of infeasible solutions and constructing new solutions from beginning. Thus the new constructed solutions can be very different from previous ones and as a consequence the change of the pheromone can be significant. In summary, even there are some numerical differences when the hybrid algorithms are compared based on minimal number of iterations and based on computation time, our conclusions of (fast) BF-ICrA are consistent.

8 Conclusions

The fast Belief Function based Inter-Criteria Analysis method, using the canonical decomposition of basic belief assignments defined on a dichotomous frame of discernment was applied, tested and analysed in this paper for two applications: (1) for evaluating the Multiple-Objective Ant Colony Optimization (MO-ACO) algorithm for Wireless Sensor Networks (WSN) deployment, and (2) for evaluating the Multiple-Objective Ant Colony Optimization (MO-ACO) algorithm for the Workforce Planning Problem (WPP).

For our first application (WSN deployment), based on the BF-ICrA outcomes we have shown a very high correlation with fast PCR6 rule for the ACO_1, ACO_3 and ACO_9 group, for the ACO_4, ACO_5 and ACO_6 group, and for the ACO_7, ACO_8 and ACO_{10} group of algorithms in Example 1 (case of size 350×350) as intuitively expected. This is because the considered ACO algorithms can solve the problem with good solution quality in Example 1. These high correlations were not observed in the other two cases for Example 2 (case of size 500×500) and 3 (case of size 700×700) because only few ACO algorithms perform actually very well for these examples. So, if we considered results in case of larger problem sizes, the BF-ICrA results show that the number of ants has the significant influence on the obtained results, as already pointed out in [22].

For our second application (WPP), based on the fast BF-ICrA results we have shown that the third variant of ACO approach, i.e. ACO_4 (ACO restart) has a quite distinct behavior with respect to the methods ACO_1, ACO_2 and ACO_3.

Acknowledgements This work is partially supported by the grant No BG05M20P001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program (2014-2020) and co-financed by the European Union through the European structural and Investment funds and the Bulgarian scientific fund by the grant DN 12/5.

References

1. Dezert, J., Tchamova, A., Han, D., Tacnet, J.-M.: Simplification of multi-criteria decision-making using inter-criteria analysis and belief functions. In: Proceedings of Fusion 2019 International Conference on Information Fusion, Ottawa, Canada, July 2–5 (2019)
2. Atanassov, K., Mavrov, D., Atanassova, V.: Intercriteria decision making: a new approach for multicriteria decision making, based on index matrices and intuitionistic fuzzy sets. Issues IFSs GNs **11**, 1–8 (2014)
3. Atanassov, K., Atanassova, V., Gluhchev, G.: InterCriteria analysis: ideas and problems. Notes IFS **21**(1), 81–88 (2015)
4. Atanassov, K., et al.: An approach to a constructive simplification of multiagent multicriteria decision making problems via intercriteria analysis. C.R. de l'Acad. Bulgare des Sci. **70**(8) (2017)
5. Atanassov, K.: Intuitionistic Fuzzy Logics. Studies in Fuzziness and Soft Computing, vol. 351. Springer, Berlin (2017) ISBN 978-3-319-48952-0
6. Dezert, J., Smarandache, F., Tchamova, A., Han, D.: Fast fusion of basic belief assignments defined on a dichotomous frame of discernment. In: Proceedings of Fusion 2020 (Online) Conference, Pretoria, South Africa (2020)
7. Dezert, J., Smarandache, F.: Canonical decomposition of dichotomous basic belief assignment. Int. J. Intell. Syst. 1–21 (2020)
8. Dezert, J., Smarandache, F.: Canonical decomposition of basic belief assignment for decision-making support. In: Proceedings of MDIS 2020 (7th international conference on Modelling and Development of Intelligent Systems), Lucian Blaga University of Sibiu, Sibiu, Romania, Oct. 22–24 (2020)
9. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
10. Dezert, J., Wang, P., Tchamova, A.: On the validity of Dempster-Shafer theory. In: Proceedings of Fusion 2012, Singapore, July 9–12 (2012)
11. Tchamova, A., Dezert, J.: On the behavior of dempster's rule of combination and the foundations of dempster-shafer theory. In: IEEE IS-2012, Sofia, Bulgaria, Sept. 6–8 (2012)
12. Dezert, J., Tchamova, A.: On the validity of Dempster's fusion rule and its interpretation as a generalization of Bayesian fusion rule. Int. J. Intell. Syst. **29**(3), 223–252 (2014)
13. Smarandache, F., Dezert, J.: On the consistency of PCR6 with the averaging rule and its application to probability estimation. In: Proceedings of Fusion 2013, Istanbul, Turkey (2013)
14. Smarandache, F., Dezert, J. (eds.): Advances and Applications of DSmT for Information Fusion, vol. 1–4. American Research Press, Santa Fe, (2004–2015). http://www.onera.fr/staff/jean-dezert?page=2
15. Smarandache, F., Dezert, J., Tacnet, J.-M.: Fusion of sources of evidence with different importances and reliabilities. In: Proceedings of Fusion 2010 Conference, Edinburgh, UK (2010)
16. https://bfasociety.org/
17. Yager, R.: On the Dempster-Shafer framework and new combination rules. Inf. Sci. **41**, 93–138 (1987)
18. Dubois, D., Prade, H.: Representation and combination of uncertainty with belief functions and possibility measures. Comput. Intell. 4 (1988)
19. Fidanova, S., Dezert, J., Tchamova, A.: Inter-criteria analysis based on belief functions for GPS surveying problems. In: Proceedings of IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA 2019), Sofia, Bulgaria, July 3–5 (2019)

20. Dezert, J., Han, D., Yin, H.: A new belief function based approach for multi-criteria decision-making support. In: Proceedings of Fusion 2016 Conference
21. Han, D., Dezert, J., Yang, Y.: Measures, new distance, of evidence based on belief intervals. In: Proceedings of Belief, Oxford, p. 2014 (2014)
22. Fidanova, S., Roeva, O.: Multi-objective ACO algorithm for WSN layout: intercriteria analysis. In: Large-Scale Scientific Computing. Springer, Berlin (2020)
23. Dorigo, M., Stutzle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
24. Fidanova, S., Roeva, O., Luque, G., Paprzycki, M.: InterCriteria analysis of different hybrid ant colony optimization algorithms for workforce planning. Recent Advances in Computational Optimization, In: Fidanova S. (eds) Recent Advances in Computational Optimization. Studies in Computational Intelligence, vol. 838, pp. 61–81 (2020)
25. Albayrak, G., Özdemir, I.: A state of art review on metaheuristic methods in time-cost trade-off problems. Int. J. Struct. Civil Eng. Res. **6**(1), 30–34 (2017)
26. Mucherino, A., Fidanova, S., Ganzha, M.: Introducing the environment in ant colony optimization, recent advances in computational optimization, studies in computational. Intelligence **655**, 147–158 (2016)
27. Roeva, O., Atanassova, V.: Cuckoo search algorithm for model parameter identification. Int. J. Bioautom. **20**(4), 483–492 (2016)
28. Tilahun, S.L., Ngnotchouye, J.M.T.: Firefly algorithm for discrete optimization problems: a survey. J. Civil Eng. **21**(2), 535–545 (2017)
29. Toimil, D., Gómes, A.: Review of metaheuristics applied to heat exchanger network design. Int. Trans. Oper. Res. **24**(1–2), 7–26 (2017)
30. Soukour, A., Devendeville, L., Lucet, C., Moukrim, A.: A Memetic algorithm for staff scheduling problem in airport security service. Expert Syst. Appl. **40**(18), 7504–7512 (2013)
31. Alba, E., Luque, G., Luna, F.: Parallel metaheuristics for workforce planning. J. Math. Modell. Algor. **6**(3), 509–528 (2007). Springer
32. Li ,G., Jiang, H., He, T.: A genetic algorithm-based decomposition approach to solve an integrated equipment-workforce-service planning problem. Omega **50**, 1–17 (2015). Elsevier
33. Grzybowska, K., Kovács, G. (2014) Sustainable supply chain - supporting tools. In: Proceedings of the 2014 Federated Conference on Computer Science and Information Systems, vol. 2, pp. 1321–1329 (2014)
34. Fidanova, S., Roeva, O., Paprzycki, M., Gepner, P.: InterCriteria analysis of ACO start strategies. In: Proceedings of the 2016 Federated Conference on Computer Science and Information Systems, pp. 547–550 (2016)
35. Fidanova, S., Luquq, G., Roeva, O., Paprzycki, M., Gepner, P.: Ant colony optimization algorithm for workforce planning. In: FedCSIS'2017, IEEE Xplorer, IEEE catalog number CFP1585N-ART, pp. 415–419 (2017)
36. Glover, F., Kochenberger, G., Laguna, M., Wubbena, T.: Selection and assignment of a skilled workforce to meet job requirements in a fixed planning period. In: MAEB'04, pp. 636–641 (2004)

Semantic Graph Queries on Linked Data in Knowledge Graphs

Jens Dörpinghaus and Andreas Stefan

Abstract Knowledge graphs have been shown to play a central role in recent knowledge mining and discovery, big data integration, especially for connecting data from different domains. Bringing structured as well as unstructured data, e.g. from scientific literature and various data sources, into a structured, comparable format is one of the key assets. KGs are usually stored in graph databases. Although a lot of research has been done on the field of query optimization, query transformation and of course in storing and retrieving large scale knowledge graphs the field of algorithmic optimization is still a major challenge and a vital factor in using graph databases. Few researchers have addressed the problem of optimizing algorithms on large scale labeled property graphs. Here, we present two optimization approaches and compare them with a naive approach of directly querying the graph database. The aim of our work is to determine limiting factors of graph databases like Neo4j and we describe a novel solution to tackle these challenges. For this, we suggest a classification schema to differ between the complexity of a problem on a graph database. In addition, we propose several other applications for graph methods within the domain of digital humanities. Here, we show how the schema helps to understand the algorithmic challenges for semantic graph queries. We evaluate other optimization approaches on a test system containing a knowledge graph derived biomedical publication data enriched with text mining data. This dense graph has more than 71 M nodes and 850 M relationships. The results are very encouraging and—depending on the problem—we were able to show a speedup of a factor between 44 and 3839.

Keywords Knowledge graphs · Graph database · Digital humanities

J. Dörpinghaus (✉)
German Center for Neurodegenerative Diseases (DZNE), Bonn, Germany
e-mail: jens.doerpinghaus@dzne.de

A. Stefan
Fraunhofer Institute for Algorithms and Scientific Computing, Schloss Birlinghoven, Sankt Augustin, Germany

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_4

1 Introduction

Although graph databases are a new field with constantly emerging technologies often missing common standards (like query languages) a lot of research has been done on the field of query optimization, query transformation and of course in storing and retrieving large scale knowledge graphs. While current state of the art systems often use RDF data models which are a collection of nested graphs and SPARQL queries the field is now driven by labeled property graphs to overcome their serious limitations. For example nodes and edges have no internal structure which does not allow complex queries like subgraph matchings or traversals and it is not possible to uniquely identify instances of relationships which have the same type, see [1].

Here, we will present research on a more general topic related to large-scale optimization in parallel and distributed computational environments: Optimization of graph algorithms using queries to communicate with a graph database backend. We present two optimization approaches and compare them with a naive approach of directly querying the graph database.

The topic of graph algorithms and their applications is widely studied in computer science and discrete mathematics. Using a graph database as data backend, graph algorithms rely on the robustness and velocity of the underlying system. This is according to our knowledge a still unconsidered topic. We will focus on a particular graph database system (Neo4j) and consider the optimization of graph algorithms an dense large scale labeled property graphs with more then 71 M nodes and 850 M edges. They are based on biomedical knowledge graphs, see [2].

Communication with the database system might either be a complex query involving heuristics (like "give me all paths from node a to b) or a simple query asking for a data set (like "give me all neighbors of node a) which are usually considered to take $O(1)$ time.

As a naive approach, we might expect that the runtime will not change using a graph database backend. If we want to find shortest paths between two nodes a and b, we can rely on a build in function. We found, that for some nodes the database backend crashed due to insufficient memory. As a second try, we can use more simple queries. For example Dijkstra's algorithm is well known to have a time complexity of $O(m + n \cdot \log(n))$ given a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, see [3]. Here, we only need to retrieve the whole set of nodes and regularly the neighborhood of nodes and the weight of edges. Although these retrievals are considered to take $O(1)$ time we have serious time problems to retrieve a dataset of 71M nodes using the Neo4j API. Using the graph database adds a factor based on a complex clew containing database efficiency, memory and computing power, connection speed and much more.

This little example above illustrates, that the usage of graph databases has serious algorithmic challenges not covered by computing complexity. The underlying challenges are related and not limited to query optimization, scaling and sharding technologies for databases and parallel algorithms. We will give an overview about this and other related work as a state of the art in the second section. After that, the

third section will give a brief overview about the background, infrastructure, data and research questions to solve. A novel, generic schema to categorize algorithms on graphs is presented in the fourth section. Here, we point at those candidates, where we need optimization approaches. The next section presents an evaluation of these optimization strategies. The sixth section introduces applications from Digital Humanities. After that, we will discuss classification approaches of these problems, new criteria and the complexity. We finish with conclusion and outlook.

2 Related Work

It is obvious, that graph databases show different query times on different situations and there is a considerable amount of literature on that topic. For example an analyses of Neo4j and the performance of queries was done by [4]. They show that there are difference in performance under different scenarios and they suggest query performance optimization for business applications. A review on storing big graphs in graph databases and their comparison is published by [1]. They conclude: "Graph data management has attracted immense research attention though it has escaped strong foundations of designing paradigms for storage and retrieval. With growth and change in data with time, the need to identify patterns and semantics becomes difficult." We will present some recent related work which underlines this statement.

A lot of research has been done with respect to analyses and optimization of graph queries, especially with focus on Cypher and Neo4j. Hölsch and Grossniklaus [5] focus on an algebraic query transformation without the usage of a relational database system to process graph data. Thakkar et al. [6] conclude, that there is a very confusing situation, it is "an unforeseen race of developing new task specific graph systems, query languages and data models, such as property graphs, key-value, wide column, resource description framework (RDF)". They focus on Gremlin, which is a graph traversal language and machine to support multiple graph systems. They suggest a graph pattern matching for Gremlin queries supporting multiple graph data models. Angles et al. [7] discuss issues of interoperability and optimization of queries between RDF and property graphs. They conclude, that more standards need to be developed. Mennicke [8] questions about the general problems of knowledge graphs: "Although graph databases are conceived schema-less, additional knowledge about the data's structure and/or semantics is beneficial in many graph database management tasks, from efficient storage, over query optimization, up to data integration." This is very plausible and we will highlight this possible pitfall in our work.

A second topic in research is the technical optimization of the database. Zhao and Han [9] address the graph query problem on large networks by decomposing shortest paths around vertex neighborhood as basic indexing unit. This was found superior to GraphQL. For Neo4j there are also several approaches. Eymer et al. [10] suggest a throughput optimization called in-graph batching which outperform standard Neo4j for large datasets. This is a similar approach to [11] who extended traditional lazy evaluation towards query batching while the application is executed.

They noticed, that usually the communication, retrieval and storing of data is a crucial factor reducing the execution time of applications. This is exactly, what we noticed in our introduction example. Other approaches have been proposed by [12] or [13].

Finally, a third way of optimization has to be mentioned. In the context of GIS graph databases [14] try to optimize the heuristic for shortest paths. While also noticing the increasing time complexity for large graphs they tried to solve the problem using filters and adjusting the algorithms. These limitations were also found by [15] while discussing the Frequent Subgraph Mining (FSM) task. Their novel TKG algorithm is also bound in size of the substructures analyzed in the graph database. One of the major drawbacks of these studies is that they focus on single problems in a very specific environment. There is still a considerable uncertainty with regard to algorithms and heuristics from a graph theoretical background when applying to graph databases.

3 Background

Using graph structures to house data has several advantages for knowledge extraction in life sciences and biological or medical research. Here, questions come from the field of exploring the mechanisms of living organisms and gaining a better understanding of underlying fundamental biological processes of life. In addition systems biology approaches, such as integrative knowledge graphs, are important as a holistic approach towards disease mechanism. In addition, pathway databases play an important role. As a basis, biomedical literature and text mining are used to build knowledge graphs, see [2]. In addition relational data from domain specific languages like BEL are widely applied to convert unstructured textual knowledge into a computable form. The BEL statements that form knowledge graphs are semantic triples that consist of concepts, functions and relationships [16]. In addition, several databases and ontologies can implicitly form a knowledge graph. For example Gene Ontology, see [17] or DrugBank, see [18] or [19] cover a large amount of relations and references to which reference other fields.

A *knowledge graph* (sometimes also called a *semantic network*) is a systematic way to connect information and data to knowledge on a more abstract level compared to language graphs. It is thus a crucial concept on the way to generate knowledge and wisdom, to search within data, information and knowledge. The context is a significant topic to generate knowledge or even wisdom. Thus, connecting knowledge graphs with context is a crucial feature.

Many authors tried to give a definition of knowledge graphs, but still a formal definition is missing, see [20]. In [21] the authors compared several definitions, but the only formal definition was related to RDF graphs which does not cover labeled property graphs. Thus, here we propose a very general definition of a knowledge graph using graph theory:

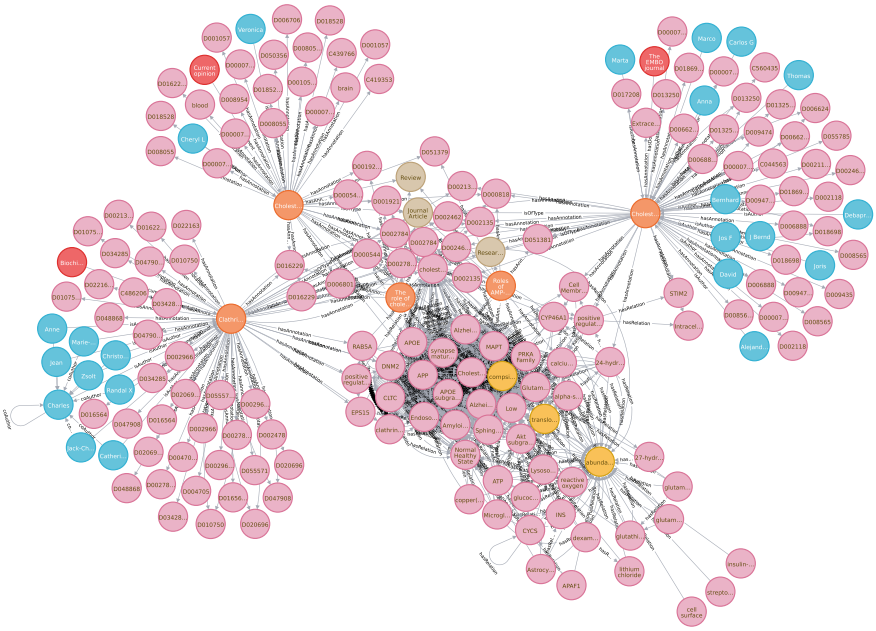


Fig. 1 A subgraph of the large scale biomedical knowledge graph. We can see three orange nodes indicating documents with their context: authors (blue), journal (red) and entities from both keywords and named entity recognition. There are many BEL relations found between single entities which in addition have relations to other documents and biological functions (yellow)

Definition 3.1 (*Knowledge Graph*) We define a knowledge graph as graph $G = (E, R)$ with entities $e \in E = \{E_1, \ldots, E_n\}$ coming from a formal structure E_i like ontologies.

The relations $r \in R$ can be ontology or layer relations (like "is related to" or "is co-Author"), thus in general we can say every formal structure E_i which is part of the data model is a subgraph of G indicating $O \subseteq G$. In addition, we allow inter-structure relations between two nodes e_1, e_2 with $e_1 \in E_1, e_2 \in E_2$ and $O_1 \neq E_2$. In more general terms, we define $R = \{R_1, \ldots, R_n\}$ as a list of either inter-structure or inner-structure relations. Both E as well as R are finite discrete spaces.

In [22] we collected 27 real world questions and queries in scientific projects to test the performance and output of the knowledge graph. We could show, that the performance of several queries was very poor and some of them even did not terminate. In order to identify limitations and understand the underlying problems, we carried on our work. The testing system is based an Neo4j and holds a dense large scale labeled property graph with more then 71 M nodes and 850 M edges. They are based on biomedical knowledge graphs as described in [2] (Fig. 1).

4 Method

Here, we propose a multi-step optimization approach towards graph queries. Usually, graph queries are executed using a Cypher query. Here, the application or the user directly communicates with the graph database. To optimize this, we suggest that an external algorithm communicates with the graph database and executes only elementary queries. With this, the queries are limited to typical questions like neighborhood, paths and relations. Since all trivial requests (like "give me this node") can usually be handled by common relational or special purpose databases, we suggest a third optimization approach, if necessary. Here, a polyglot persistence approach uses other data sources to execute trivial queries. See Fig. 2 for an illustration.

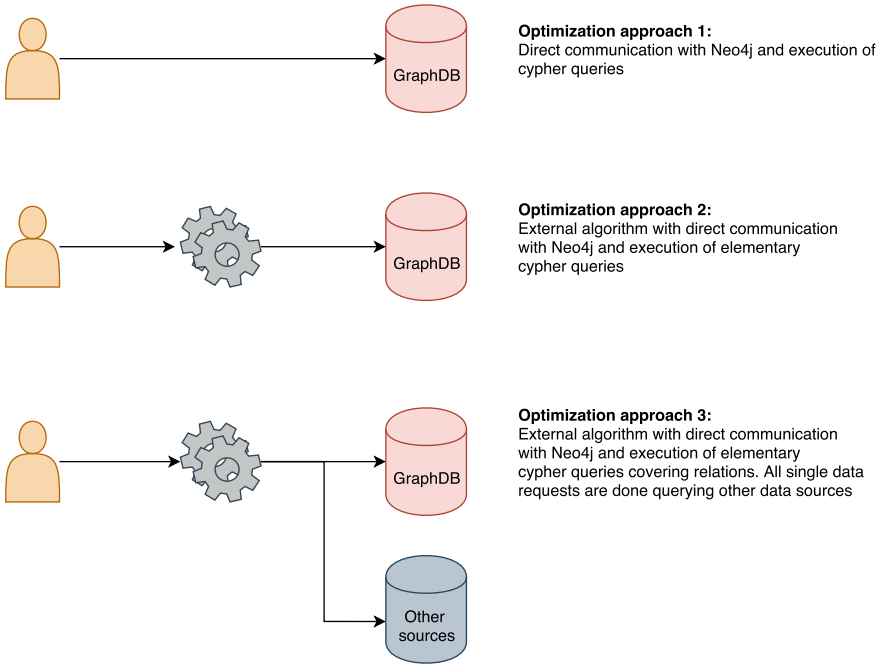


Fig. 2 An overview of the optimization approaches discussed in this paper. The first approach contains the basic Cypher query, the second approach transfers the algorithm to a different system. The third approach relies on a polyglot persistence architecture and excludes all time-consuming queries that can be answered by a key-value store

4.1 Pathfinding

In [22] we introduced a large set of queries and categorized them according to the schema discussed in Sect. 7. We will start with those problems using in general both locale as well as global structures in the graph. A problem with a very poor performance was graph navigation and pathfinding. These include Regular Path Queries (RPQ, see [23]) (problems 2, 11, 14, 16, 17, 19, 21) and finding shortest paths (problems 4, 12). Since the problems of retrieving a single or all shortest paths are quite similar, we will discuss both of them here.

Queries 4 and 12 are both a typical shortest path problem: *What is the shortest way between {Entity1} and {Entity2} and what is on that way?* and *How far apart are {document1} and {document2}?* Thus both problems can be solved using Cypher:

(Q4) `match (entity1:Entity {preferredLabel: "axonal transport"}), (entity2:Entity {preferredLabel: "LRP3"}) call algo.shortestPath.stream(entity1, entity2) yield nodeId return algo.asNode(nodeId).`

(Q12) `match (doc1:Document {documentID: "PMID:16160056"}), (doc2:Document {documentID: "PMID:16160050"}) call algo.shortestPath.stream(doc1,doc2) yield nodeId return algo.asNode(nodeId).`

Both queries rely on the function `shortestPath` available in Neo4j. Both Bellman–Ford and Dijkstra's algorithm are known to solve this problem for weighted graphs. For unweighted graphs a modified Breadth-first search will solve this issue in $O(E + V)$ [24]. Other algorithms like Dijkstra's should be faster, for example using binary heaps the time complexity is $O(m + n \cdot \log(n))$ given a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, see [3]. According to Neo4j documentation, the build in function `shortestPath` uses Dijkstra's algorithm.[1]

With Algorithm 1 we suggest a BFS-approach to tackle the shortest-path problems. Given both a starting node s and an ending node e, the only communication with the graph database is done in line 18. Here, the neighborhood of a node is retrieved.

This algorithm implements the optimization approach 2. Since no other data sources are needed, optimization approach 3 will not improve this query.

4.2 CRPQ

Several questions introduced in [22] are conjunctive regular path queries (CRPQ, see [25]). These are pattern matching problems using locale structures within the graph. Some of them are quite simple. For example query 15—*How many sources are there for the statements of a contradictory BEL statement?*—can be easily translated into Cypher:

[1] See https://neo4j.com/docs/graph-algorithms/current/labs-algorithms/shortest-path/.

Algorithm 1 GRAPH-BFS

Require: two nodes $s, e \in V$
Ensure: shortest path $p = [s, \ldots, e]$

```
Q = []
discovered = [s]
Parent =
Q.append(s)
while len(Q) > 0 do
  v = Q.pop(0)
  if getNode(v) == e then
    x = v
    path = [v]
    while Parent[x]! = s do
      x = Parent[x]
      path.append(x)
    end while
    x = Parent[x]
    path.append(x)
    returnpath
  end if
  N = getNeighbours(v)
  for winN do
    if w not in discovered then
      discovered.append(w)
      Parent[w] = v
      Q.append(w)
    end if
  end for
end while
return d with max (pd)
```

(Q15) `match (e1:Entity) -[r1:hasRelation {function: "increases"}]-> (e2:Entity), (e1) -[r2:hasRelation {function: "decreases"}]-> (e2) return distinct e1.preferredLabel, e2.preferredLabel, count(r1) as 'increases', count(r2) as 'decreases' order by count(r1) desc.`

This query matches two contradicting relations, their numbers and returns a decreasing sorted list. More complex is the example query 1: *Which author was the first to state that {Entity1} has an enhancing effect on {Entity2}?* A Cypher query solving this uses several node attributes, for example the publication date to sort the result set:

(Q1) `match (n:Entity preferredLabel: "APP") -[r:hasRelation function: "increases"]-> (m:Entity preferredLabel: "gamma Secretase Complex"), (doc:Document documentID: r.context) <-[r2:isAuthor]- (author:Author) return doc, author order by doc.publicationDate limit 1'.`

As a first optimization approach denoted by opt1 we exclude the sorting functions from the queries and do this manually. This leads to the following two queries:

(Q1-1) `match (n:Entity preferredLabel: "APP")-[r:has Relation function: "increases"]->(m:Entity preferred Label: "gamma Secretase Complex") return n,r,m.`

(Q15-1) `match (e1:Entity) -[r1:hasRelation function: "increases"]-> (e2:Entity), (e1) -[r2:hasRelation function: "decreases"]-> (e2) return distinct e1. preferredLabel, e2.preferredLabel.`

The algorithm for query 1 can be found in Algorithm 2, the algorithm for query 15 in Algorithm 3. As we can see, query 1 is more complex, since it includes the retrieval of node attributes, the publication data. Both algorithms include the sorting of lists.

Algorithm 2 QUERY1-OPT1

Require: Documents $D = \{d_1, \ldots, d_n\}$ obtained from query (Q1-1)
Ensure: Document d
$pd = []$
2: **for** every $d \in D$ **do**
pd.add (d,d.publicationdate)
4: **end for**
return d with max (pd)

Algorithm 3 QUERY15-OPT1

Require: Data points $T = \{t_1, \ldots, t_n\}$ with $t_i = \{e1_i, e2_i, inc_i, dec_i\}$ obtained from query (Q15-1)
Ensure: Sorted data points T
return sort(T)

The second optimization approach can only be applied to query 1. Here, we try to retrieve the node attributes from a dedicated information system. This is related to the polyglot persistence approach introduced in [22]. Here, we suggest to retrieve this value direct from the SCAIView API.

(Q1-2) `match (n:Entity preferredLabel: "APP")-[r:has Relation function: "increases"]->(m:Entity preferred Label: "gamma Secretase Complex") return n,r,m.`

Here, algorithm QUERY1-OPT2 will use a different function to add the publicationdate in line 3.

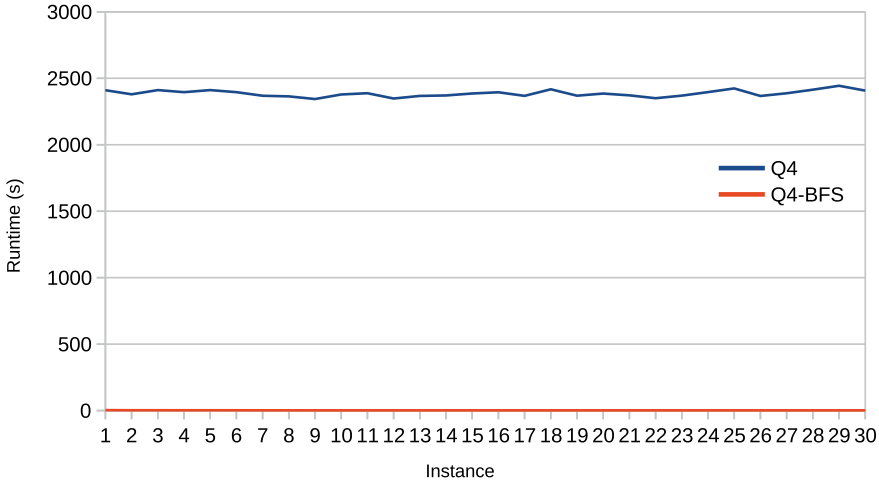


Fig. 3 Results for query 4 QUERY4 (average runtime 2390.44 s) and the optimization approach 1 GRAPH-BFS (average runtime 1.65 s). The speedup factor is 1453

5 Evaluation

We evaluate our optimization approaches on a test system containing a knowledge graph derived biomedical publication data enriched with text mining data and domain specific language data using BEL, see [22]. This dense graph has more than 71 M nodes and 850 M relationships.

The testing system run Neo4j Community 3.5.8. on a server with 16 Intel Xeon CPUs with 3 GHz and 128 GB main memory. We applied several approaches described in the chapter "Performance" in the Neo4j Operations Manual.[2]

5.1 Pathfinding

Both queries 4 and 12 are pathfinding problems. To retrieve the shortest path, we suggested the execution of a Cypher query using the build in `shortestPath` algorithm. Applying optimization strategy 1, we suggest the usage of a BFS-approach called GRAPH-BFS.

Contrary to expectations, build in Dijkstra's algorithm performs very poor. The runtime lay between 40 and 60 min, the average runtime was 2390.44 s. In contrast the BFS-approach had a runtime of 1–2 s, the average runtime was 1.65 s. This is a speedup factor of 1453, see Fig. 3.

[2] See https://neo4j.com/docs/operations-manual/current/performance/.

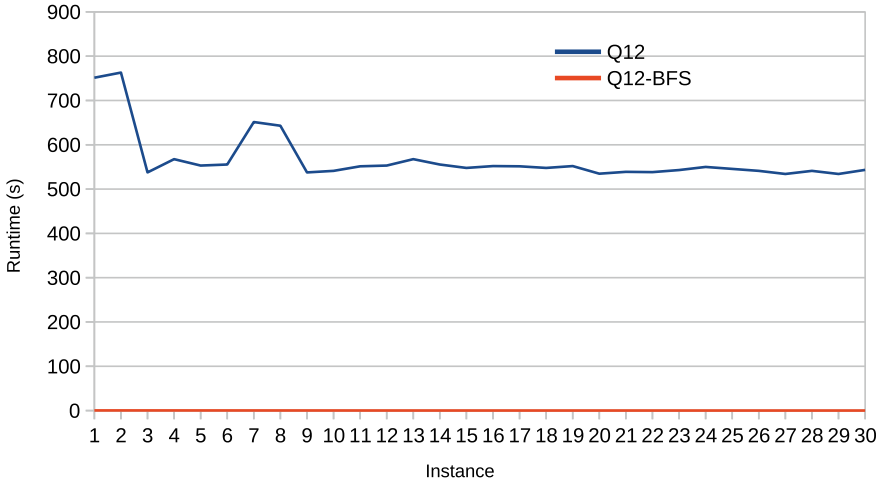


Fig. 4 Results for query 12 QUERY12 (average runtime 567.44 s) and the optimization approach 1 GRAPH-BFS (average runtime 0.14 s). The speedup factor is 3838.77

These results could also be reproduced on Query 12, see Fig. 4. The average runtime of `shortestPath` is 567.44 s, approximating 10 min. The average runtime of the BFS-approach is 0.14 s. This is a speedup factor of 3838.77, see Fig. 4.

These results highlighted that the `shortestPath` function cannot be used for large scale knowledge graphs due to the runtime. Unexpectedly, the simple BFS-approach utilizing our first optimization strategy decreases the runtime nearly by the factor 3840. Further analysis showed that the speedup is highly influenced by node degree. Nevertheless, `shortestPath` is unacceptable for information systems with a user frontend.

5.2 CRPQ

We had a more simple query (15) and a more complex query (1). Regarding QUERY15, we could only implement our first optimization approach QUERY15-OPT1. Figure 5 presents the runtime data. The average runtime of QUERY15 is 8.6 s, the average runtime of QUERY15-OPT1 is 8.4 s. As we can see, there is no real advantage in applying the optimization approach here. In general both heuristics are competitive, while the simple Cypher query has some situations where it is significantly slower. Although no significant differences were found, the optimization approach shows a rather constant runtime.

The most striking results are obtained with more complex queries. The situation changes significantly when analyzing query 1. Here, the Cypher query QUERY1 usually has an execution time of about 7 or 8 min, the average runtime is 364.45 s.

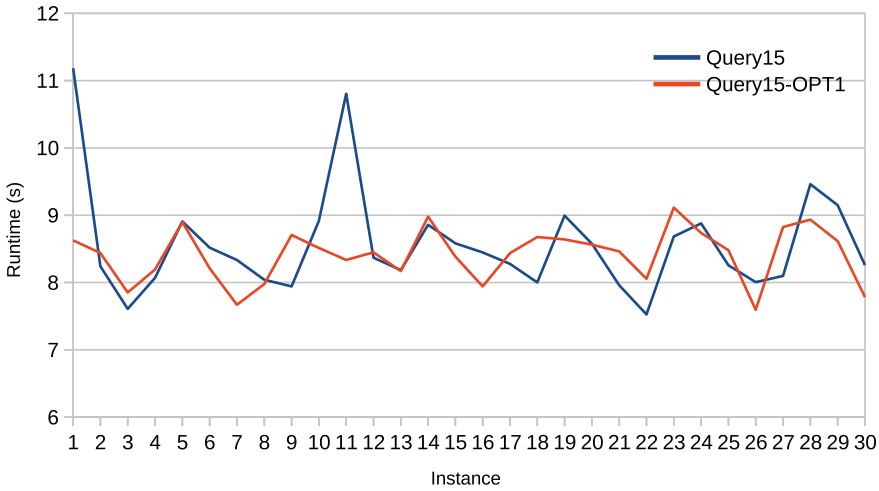


Fig. 5 Results for query 15 QUERY15 (average runtime 8.6 s) and the optimization approach 1 QUERY15-OPT1 (average runtime 8.4 s)

Using the optimization approach 1, the execution time of QUERY1-OPT1 reduces to 1–2 min, the average runtime is 80.2 s. Thus, the runtime decreases by the factor 4.5. Using an polyglot persistence approach and querying SCAIView for the metadata, the execution time of QUERY1-OPT2 once again decreases to more or less 10 s, in average 9.6 s. Here, the runtime decreases by the factor 9.6 compared with QUERY1-OPT1 and by the factor 43.8 compared with QUERY1, see Fig. 6.

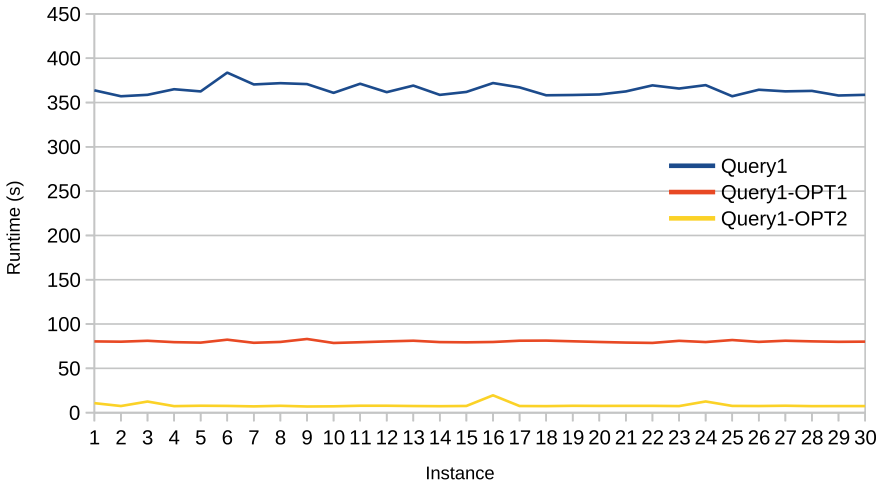


Fig. 6 Results for query 1 QUERY1 (average runtime 364.45 s) and the optimization approaches 1 QUERY1-OPT1 (average runtime 80.2 s) and 2 QUERY1-OPT2 (average runtime 9.6 s). In total the speedup factor is 43.8

It is important to note, that simple queries like Q15 cannot be improved very easy. Graph databases are highly optimized to retrieve relations. But our technique shows a clear advantage over simple Cypher queries when multiple relations are queried, functions for sorting or other purposes are called and especially when single nodes or edges are called to retrieve metadata. Neo4j shows no good performance when used as a key-value store.

6 Applications from Digital Humanities: Centrality Measures

Network-based approaches have been used in digital humanities, social sciences and history for a long time to examine social structures of human interaction. The aim of Social Network Analyses (SNA) is to convert human interaction into measurable and countable networks. It developed within social sciences and is under constant discussion in other sciences like history and theology as well, see the work of Rollinger [26], or Collar [27]. It is of some interest for theology, as Duling stated out: "interest in SNA by Biblical scholars has been sporadic, but steady, and is apparently growing" [28].

Jacob Moreno was the first person talking about social networks. He did research on relationships within networks. In this early phase there was a collaboration between social sciences and ethnology. The breakthrough of SNA was during the 1970s in the USA. Today scientists mainly focus on quantitative and structural anal-

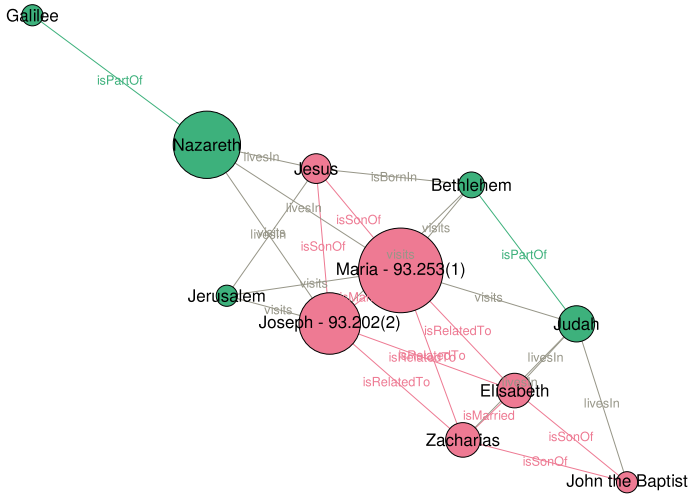


Fig. 7 The social network representation of Lk 1:1-2:22. Red nodes describe person, green nodes locations. The edges describe different relations. The node size reflects the betweenness centrality of each node

yses and Actor Network Theory (ANT). The research question is mainly how information flow withing the network and how non-static networks change. In historical research the idea of SNA spread from the US to Europe within the last decades. Here, scientists make extensive use of the Proximal Point Analyses (PPA), small-World order Complexity Theory, Relational Space, Closeness and Betweenness Centrality measures within the SNA.

SNA—already a knowledge graph per definition—can be easily extended to a "full" knowledge graph in combining literature sources and other knowledge entities. Here, we will present an example from theology and the algorithmic output of this approach.

To visualize and analyze early Christianity in the New Testament we propose the Social Network Analyses (SNA) in order to determine the concept of space—both social and topographical. This methods requires an exegetical approach described by Luke's historical work, Paul's letters and all other historical sources available for early Christianity.

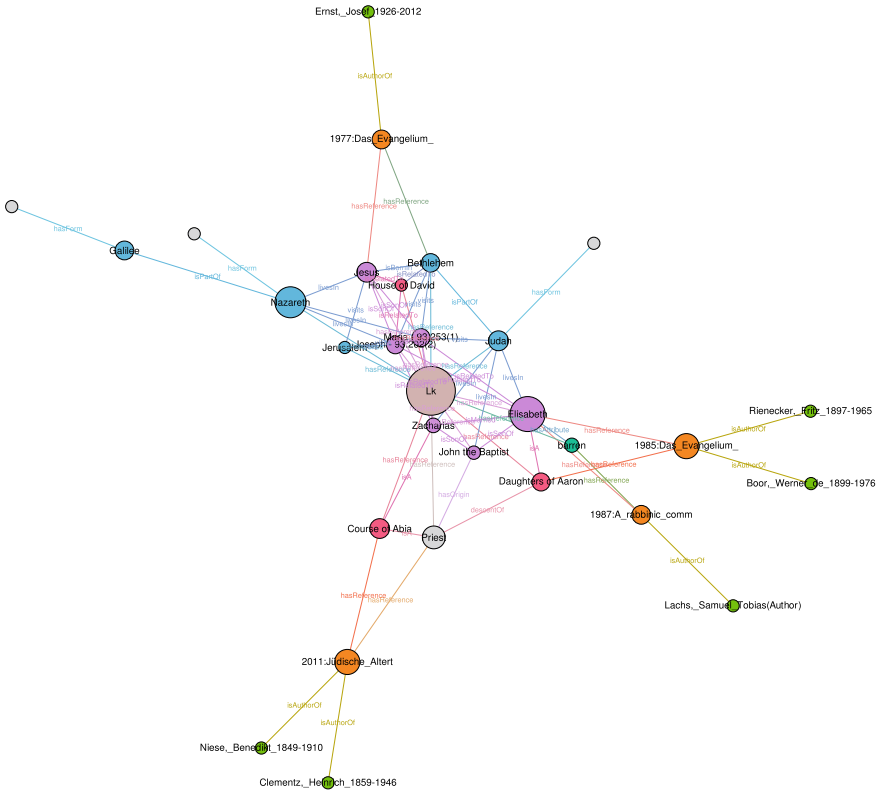


Fig. 8 The knowledge graph representation of Lk 1:1-2:22. Purple nodes describe person, blue nodes locations, green, grey and red nodes different entities, orange nodes literature references and green nodes authors. The edges describe different relations. The node size reflects the betweenness centrality of each node

First, we will discuss a short example of a knowledge graph extension of a social network. In Fig. 7 we can see a social network representation of Lk 1:1-2:22. There are two different sets of entities. E_1 describes persons, E_2 locations. The edges describe different relations between both entities from the same set as well as between entities from different sets. A more generalized network or knowledge graph can be found in Fig. 8. Here, every node and edge has some sort of source or provenance. The brown node "Lk" refers to the Gospel of Luke. If we find references or sources in scientific literature, they form orange nodes. There are also some other concepts or entities that can be seen as context, for example being "barren" or being a "priest". These concepts will help to detect previously hidden knowledge with the help of computer science.

Building a more complex network was proposed in [29]. Figure 9 shows an illustration of the social network described in Acts 1–12. The node's size describes the betweenness centrality measure for each actor. Here, the algorithmic challenge is to compute distance measures. These problems are in $\mathcal{P}$. The instance graph is quite small and thus a solution can be computed in very short time.

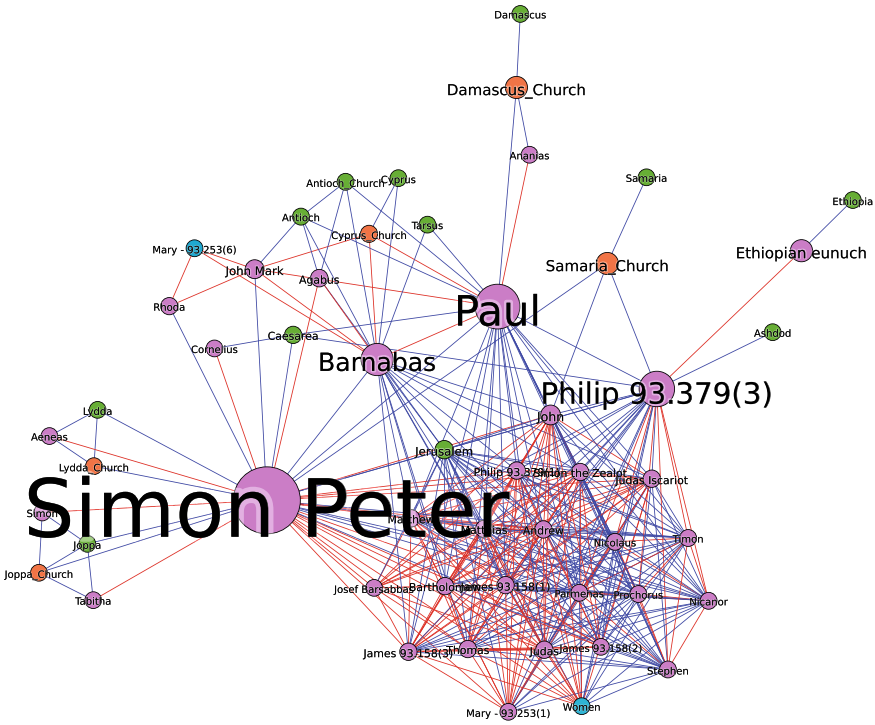


Fig. 9 The social network of Acts 1–12. Weak ties are represented by blue edges, strong ones by red. The size of a node indicates its betweenness centrality. The node's color reflects their group affiliation. Purple denotes persons, green denotes Places, Orange communities, Blue groups of people

7 Classification of Problems

There seems to be no generally established procedure for categorizing graph-based queries. What we know about graph queries is largely based on six sources that categorize graph queries or describe them according to different criteria. The contents of this work and the results of the criteria are presented in this section.

Barceló Baeza [30] examines various theoretical classes of graph query languages with respect to the possible expressions and the complexity of evaluating queries. However, the study is not based on the property graph model, but on a simpler model with a finite directed graph with edge labels. In their analysis they show that for current graph databases, including Neo4j, there is a lack of a language with clear syntax and semantics. They claim that this is a difficulty to evaluate the expressiveness and computational effort of possible queries.

(1) Angles [31] describe graph queries that are considered relevant on the basis of the author's literature research and can be divided into the following four categories:

- Adjacency queries
  Adjacency queries check whether two nodes are connected or in the *k neighborhood* of each other.
- Reachability queries
  Accessibility queries check whether a node can be reached via a fixed-length path or via a simple regular path and which is the shortest path between the nodes.
- Pattern Matching queries
  Pattern matching consists of finding all subgraphs of a graph that are isomorphic to a pattern graph.
- Summarization queries
  These types of queries are based on functions that allow the results of the queries to be summarized, usually returning a single value. These include functions such as average, number, maximum, etc. They also include functions for calculating properties of the graph and its elements such as the degree of a node, the minimum, maximum and average degrees in the graph, the length of a path, the distance between two nodes, the diameter of the graph, etc.

(2) Angles et al. [23] divide graph queries into two basic functions: *Graph Patterns*, where a pattern structured as a graph is searched in the database, and *Graph Navigation*, which should find paths of any length. The graph pattern queries can be further restricted by projection, union and difference. The result of a Graph Pattern query is a set of all mappings of variables from the query to constants in the database.

The simplest query in the class of Graph Navigation Queries is whether a certain path exists in the graph. This can be extended by additional restrictions, for example, by allowing only certain edge labels. To do this, a path query can be described in general terms as $P = x \longrightarrow \alpha y$, where α specifies the restrictions. The endpoints x and y can be variables or specific nodes. The best known formalism for representing α is *regular expressions*. Regular expressions allow the concatenation of paths and the application of a union or disjunction of paths. Path queries specified with regular expressions are commonly referred to as *Regular Path Queries (RPQ)*. Angles

et al. [23] provide information on the complexity of evaluating RPQs to determine whether a path exists. However, the complexity information for RPQs cannot simply be applied to Cypher. In addition, they show that several open questions regarding complexity or the graph query language Cypher remain. In contrast to SPARQL, the semantics and complexity of Cypher has not yet been investigated due to the lack of theoretical formalization, see [23, 25].

(3) Wood [25] describe different classes of queries for several graph query languages, as well as several core functionalities supported by the graph query languages. They also discuss the expressiveness and complexity of query evaluation. Unfortunately, Cypher is not described as a graph query language. The author divides the queries into the following categories:

- *CQ* (conjunctive query)
  A sample query of this type looks for documents that have both the *PublicationType* `Journal Article` and `Review`.
- *RPQ* (regular path query)
  A search is made for a node pair (x, y) so that a path exists between x and y, with the sequence of edge labels following a given pattern. The given pattern is described by a regular expression.
- *CRPQ* (conjunctive regular path queries)
  *CQ*s and *RPQ*s can be combined to form the class *CRPQ*. According to the author, this class serves as a basis for several graph query languages. However, this class is not sufficient for problems where relationships between paths need to be specified.
- *ECRPQ* (extended conjunctive regular path query
  This class extends the *CRPQ*s by the possibility to specify path variables or to allow paths as output of a query.

In addition [25] examine functionalities of graph query languages. They are divided into the following categories:

- Subgraph Matching
  It searches for subgraphs in a graph. This is a *CQ*.
- Find connected nodes by path
  Determining accessibility between nodes in a graph is a graph query that is supported in many graph query languages. The *RPQ*s class includes queries that return all node pairs from a graph that are connected by a path that matches a regular expression.
- Compare and return paths
  It specifies relationships between paths and searches for paths that connect two nodes to find connections in linked data. By providing these two functions, the class of extended *CRPQ*s (*ECRPQ*s) is created.
- Aggregation
  Determining different properties of graphs requires a calculation that goes beyond matching and finding paths. Such properties are for example the determination of node degrees.

(4) Both [32, 33] consider queries with property graphs and name among others Cypher and Gremlin as important graph query languages. These sources name these categories of graph queries:

- k-hop Queries
  According to the authors, these queries are most common in practice. They include queries such as *find node*, *find the node's neighbors (1-hop query)*, *find edges in multiple hops*, and *get attribute values*.
- subgraph and supergraph queries
- width search / depth search
- Seeking and shortcuts
- Search for strongly connected components
- Regular Path Queries

(5) In [34], queries and graph algorithms are described and subdivided according to different properties. On the one hand, the authors subdivide the queries according to *graph pattern*-based queries for local analysis of the data and according to *graph algorithms*, which often analyze globally and iteratively. Local queries only look at a specific section of the graph like a start node and the surrounding subgraph. This type of query is often used for transactions and pattern-based queries. Graph algorithms typically search for global structures. The algorithm takes the entire graph as input and returns an enriched graph or an aggregated value.

The authors divide different graph algorithms into the three categories *Pathfinding*, *Centrality* and *Community Detection*. The book describes several graph algorithms and assigns them to the categories:

- pathfinding
  - Shortest Paths
  - All Pair Shortest Path
  - Minimum Spanning Tree
  - random walk
- Centrality
  - Degree Centrality
  - Closeness Centrality
  - Betweenness Centrality
  - page rank
- Community Detection
  - Triangle Count
  - (Strongly) Connected Components
  - Label Propagation
  - Louvain Modularity

7.1 New Criteria

In order to categorize graph queries, we introduce new criteria, which were found relevant for the use case evaluated for our knowledge graph:

- Accessing attributes
  How many attributes must be considered when executing the query? Accessing attributes requires reading an additional file and therefore requires more processing power and access time. In Sect. 5 we will proof, that data stored in attributes will significantly slow down queries.
- Data type of attributes
  What data types are accessed in queries? We expect, that this also influences the runtime.
- Node and edge types to be considered
  Which node and edge types must be considered in the query? Is it only a small subset or is the majority of the types required? Is it possible to decide for all queries whether and which node and edge types can be exported as subgraphs?
- Entry point
  Does the query rely on a unique node specified for the query (e.g. as a starting point for the search), or is there a general search for pattern between nodes?

Various approaches have been proposed in literature, but we can examine connections and a hierarchy. In the next step, we merge and cluster these approaches in order to create a categorization scheme for graph queries. This is shown in Fig. 10.

The schema divides the categories for graph queries into *local structures* and *local & global* structures (according to literature source (5)). The second structure category is called *local & global*, because some of the graph algorithms can act locally by specifying a start node or a subgraph. Furthermore, some categories, such as *CRPQ* or *ECRPQ*, were identified as subcategory of other categories. This is illustrated by the hierarchical structure of the categorization scheme. The category

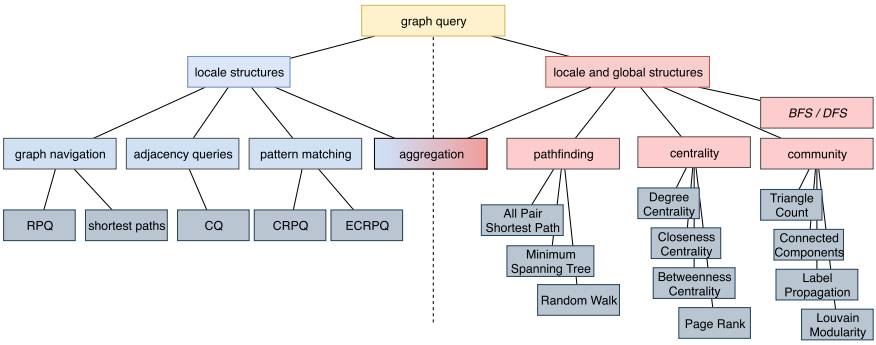


Fig. 10 An overview of the categories for graph queries unified from literature sources. These categories give a first overwiev and a categorization scheme for graph queries and their complexity

Aggregation belongs to graph queries that search for both local and global structures. For example, the category *Aggregation* can include questions such as "What is the degree of node A?" or "What is the average of the graph?", the former referring to local and the latter to global structures.

7.2 Complexity

While some problems are known to have solutions running in polynomial time (like pathfinding), others are well known to be quite hard. For example graph navigation and pattern matching are more complex. RPQ is known to be PSPACE-complete, CRPQ and ECRPQ are EXPSPACE-complete, see [35].

Centrality measures for knowledge graphs are also quite complex. While several efficient algorithms have been proposed, see [36], and some more specific problems are known to be $\mathcal{NP}$-hard. For example Group Closeness Maximization (GCM), see [37] or Maximum Betweenness Centrality, see [38].

8 Conclusion and Outlook

In this paper we presented two new approaches for query optimization on large scale knowledge graphs using graph databases. Knowledge graphs have been shown to play an important role in recent knowledge mining and discovery. A knowledge graph (sometimes also called a semantic network) is a systematic way to connect information and data to knowledge on a more abstract level compared to language graphs.

We used three approaches to compare our optimization strategies to state-ofthe-art Cypher queries. Our goal was to reach the best optimization level without changing the underlying graph database. We believe this solution will aid researchers without a technological background to effectively improve their queries. Our experiments showed that the proposed optimization strategies can effectively improve the performance by excluding those parts of queries with the highest runtime. Especially the retrieval of single entities like nodes and edges, but also the usage of functions like sorting or shortest paths have been detected for decreasing the execution time significantly.

Graph databases are highly efficient and optimized for storing and retrieving relations between data points. Thus, we propose to review graph queries carefully and check, if heuristics can be used to merge those parts of a query that are very fast in graph databases. Thus it is an important step to provide a deeper understanding of the underlying graph structures. We could show that most graph queries categorized as locale structures cannot be executed efficiently out of the box: graph navigation and pattern matching. Only adjacency queries seem to perform very good.

Although this is a good step towards a better understanding of the underlying problem field, it does not help to find a general solutions to optimize graph queries. Improving the runtime of graph queries needs a careful understanding and improving of the heuristics.

Our future work includes optimization approaches for federated queries on multiple data sources and better understanding of those cases, where optimization approaches are feasible and lead to a significant improvement of execution time. In addition we plan to evaluate our results with other graph databases like OrientDB.

References

1. Desai, M., Mehta, R.G., Rana, D.P.: Issues and challenges in big graph modelling for smart city: an extensive survey. Int. J. Comput. Intell. & IoT **1**(1) (2018)
2. Dörpinghaus, J., Stefan, A.: Knowledge extraction and applications utilizing context data in knowledge graphs. In: 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), pp. 265–272. IEEE (2019)
3. Johnson, D.B.: Efficient algorithms for shortest paths in sparse networks. J. ACM (JACM) **24**(1), 1–13 (1977)
4. Huang, H., Dong, Z.: Research on architecture and query performance based on distributed graph database neo4j. In: 2013 3rd International Conference on Consumer Electronics, Communications and Networks, pp. 533–536. IEEE (2013)
5. Hölsch, J., Grossniklaus, M.: An algebra and equivalences to transform graph patterns in neo4j. In: EDBT/ICDT 2016 Workshops: EDBT Workshop on Querying Graph Structured Data (GraphQ) (2016)
6. Thakkar, H., Punjani, D., Auer, S., Vidal, M.-E.: Towards an integrated graph algebra for graph pattern matching with gremlin. In: International Conference on Database and Expert Systems Applications, pp. 81–91. Springer (2017)
7. Angles, R., Thakkar, H., Tomaszuk, D.: Rdf and property graphs interoperability: status and issues. In: Proceedings of the 13th Alberto Mendelzon International Workshop on Foundations of Data Management, Asunción, Paraguay (2019)
8. Mennicke, S.: Modal schema graphs for graph databases. In: International Conference on Conceptual Modeling, pp. 498–512. Springer (2019)
9. Zhao, P., Han, J.: On graph query optimization in large networks. Proc. VLDB Endow. **3**(1–2), 340–351 (2010)
10. Eymer, J., Dexter, P., Liu, Y.D.: Toward lazy evaluation in a graph database. SPLASH (2019)
11. Cheung, A., Madden, S., Solar-Lezama, A.: Sloth: being lazy is a virtue (when issuing database queries). ACM Trans. Database Syst. (ToDS) **41**(2), 8 (2016)
12. Mathew, A.B.: Efficient query retrieval from social data in neo4j using lindex. KSII Trans. Internet Inf. Syst. **12**(5) (2018)
13. Cabrera, W., Ordonez, C.: Scalable parallel graph algorithms with matrix-vector multiplication evaluated with queries. Distrib. Parallel Databases **35**(3–4), 335–362 (2017)
14. Wu, X., Deng, S.: Research on optimizing strategy of database-oriented gis graph database query. In: 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), pp. 305–309. IEEE (2018)
15. Fournier-Viger, P., Cheng, C., Chuan Wei, L.J., Yun, U., Kiran, R.U.: Tkg: efficient mining of top-k frequent subgraphs. In: Big Data Analytics: 7th International Conference, BDA 2019, Ahmedabad, India, December 17–20, 2019, Proceedings, vol. 11932, p. 209. Springer Nature (2019)

16. Fluck, J., Klenner, A., Madan, S., Ansari, S., Bobic, T., Hoeng, J., Hofmann-Apitius, M., Peitsch, M.: Bel networks derived from qualitative translations of bionlp shared task annotations. In: Proceedings of the 2013 Workshop on Biomedical Natural Language Processing, pp. 80–88 (2013)
17. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., et al.: Gene ontology: tool for the unification of biology. Nature Gen. **25**(1), 25 (2000)
18. Wishart, D.S., Feunang, Y.D., Guo, A.C., Lo, E.J., Marcu, A., Grant, J.R., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., et al.: Drugbank 5.0: a major update to the drugbank database for 2018. Nucl. Acids Res. **46**(D1), D1074–D1082 (2017)
19. Khan, K., Benfenati, E., Roy, K.: Consensus qsar modeling of toxicity of pharmaceuticals to different aquatic organisms: ranking and prioritization of the drugbank database compounds. Ecotoxicol. Environ. Safety **168**, 287–297 (2019)
20. Fensel, D., Şimşek, U., Angele, K., Huaman, E., Kärle, E., Panasiuk, O., Toma, I., Umbrich, J., Wahler, A.: Introduction: What Is a Knowledge Graph?, pp. 1–10. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-37439-6_1
21. Ehrlinger, L., Wöß, W.: Towards a definition of knowledge graphs. SEMANTiCS (Posters, Demos, SuCCESS) (48) (2016)
22. Dörpinghaus, J., Stefan, A., Schultz, B., Jacobs, M.: Towards context in large scale biomedical knowledge graphs (2020). arXiv:2001.08392
23. Angles, R., Arenas, M., Barceló, P., Hogan, A., Reutter, J., Vrgoč, D.: Foundations of modern query languages for graph databases. ACM Comput. Surv. **50**(5), 68:1–68:40 (2017). http://doi.acm.org/10.1145/3104031
24. Aziz, A., Prakash, A.: Algorithms for interviews: a problem solving approach (2010). https://algorithmsforinterviews.com/
25. Wood, P.T.: Query languages for graph databases. SIGMOD Rec. **41**(1), 50–60 (2012). http://doi.acm.org/10.1145/2206869.2206879
26. Rollinger, C.: Amicitia sanctissime colenda. Freundschaft und soziale Netzwerke in der Späten Republik (2014)
27. Collar, A.: Religious Networks in the Roman Empire. Cambridge University Press, Cambridge (2013)
28. Duling, D.C.: Paul's aegean network: the strength of strong ties. Biblical Theol. Bull. **43**(3), 135–154 (2013)
29. Dörpinghaus, J.: Soziale Netzwerke im frühen Christentum nach der Darstellung in Apg 1–12 (2020). http://uir.unisa.ac.za/handle/10500/26609
30. Barceló Baeza, P.: Querying graph databases. In: Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (2013)
31. Angles, R.: A comparison of current graph database models. In: 2012 IEEE 28th International Conference on Data Engineering Workshops, pp. 171–177 (2012)
32. Pokorný, J.: Functional querying in graph databases. Vietnam J. Comput. Sci. **5**(2), 95–105 (2018). https://doi.org/10.1007/s40595-017-0104-6
33. Pokorny, J.: Graph databases: their power and limitations. In: Saeed, K., Homenda, W. (eds.) Computer Information Systems and Industrial Management, pp. 58–69. Springer International Publishing, Cham (2015)
34. Needham, M., Hodler, A.E.: Graph Algorithms. O'Reilly Media, Inc., Sebastopol (2019)
35. Bonifati, A., Dumbrava, S.: Graph queries: from theory to practice. ACM SIGMOD Record **47**(4), 5–16 (2019)
36. Grando, F., Noble, D., Lamb, L.C.: An analysis of centrality measures for complex and social networks. In: IEEE Global Communications Conference (GLOBECOM), vol. 2016, pp. 1–6. IEEE (2016)
37. Chen, C., Wang, W., Wang, X.: Efficient maximum closeness centrality group identification. In: Australasian Database Conference, pp. 43–55. Springer (2016)
38. Fink, M., Spoerhase, J.: Maximum betweenness centrality: approximability and tractable cases. In: International Workshop on Algorithms and Computation, pp. 9–20. Springer (2011)

Online Single-Machine Scheduling via Reinforcement Learning

Yuanyuan Li, Edoardo Fadda, Daniele Manerba, Mina Roohnavazfar, Roberto Tadei, and Olivier Terzo

Abstract Online scheduling has been an attractive field of research for over three decades. Some recent developments suggest that Reinforcement Learning (RL) techniques can effectively deal with online scheduling issues. Driven by an industrial application, in this paper we apply four of the most important RL techniques, namely *Q-learning*, *Sarsa*, *Watkins's Q(λ)*, and *Sarsa(λ)*, to the online single-machine scheduling problem. Our main goal is to provide insights into how such techniques perform in the scheduling process. We will consider the minimization of two different and widely used objective functions: the total tardiness and the total earliness and tardiness of the jobs. The computational experiments show that Watkins's Q(λ) performs best in minimizing the total tardiness. At the same time, it seems that the RL approaches are not very effective in minimizing the total earliness and tardiness over large time horizons.

Keywords Single-machine scheduling · Metaheuristic · Reinforcement learning

Y. Li
ESSEC Business School, 95000 Cergy, France
e-mail: yuanyuan.li@essec.edu

E. Fadda (✉) · M. Roohnavazfar · R. Tadei
Politecnico di Torino, Department of Control and Computer Engineering, corso Duca degli Abruzzi 24, 10129 Torino, Italy
e-mail: edoardo.fadda@polito.it

M. Roohnavazfar
e-mail: mina.roohnavazfar@polito.it

R. Tadei
e-mail: roberto.tadei@polito.it

D. Manerba
Department of Information Engineering, University of Brescia, 25123 Brescia, Italy
e-mail: daniele.manerba@unibs.it

O. Terzo
LINKS Foundation - Advanced Computing and Applications, via Pier Carlo Boggio 61, 10138 Torino, Italy
e-mail: olivier.terzo@linksfoundation.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_5

1 Introduction

Production scheduling is one of the most important aspects to address in many manufacturing companies (see [3]). The optimization problems arising within production scheduling can be of *static* or *dynamic* type (see [17]). In contrast with the static case, in which specifications and requirements are fully and deterministically known in advance, in the dynamic one, additional information (e.g., new orders, changes of available resources) may arrive during the production process itself. In this paper, we will consider the latter case, commonly called *online scheduling*, mainly fostered by our experience on an industrial project (Plastic and Rubber 4.0[1]) in which frequent occurrences of unexpected events call for more dynamic and flexible scheduling (see [22]).

We will mainly focus on online single-machine scheduling problems with release dates, where preemption is allowed. Let us consider a set $\mathcal{J}$ of jobs that are released over time. As soon as a job arrives, it is added to the end of a waiting queue. For each job $j \in \mathcal{J}$, let d_j be its due date and c_j its completion time. A job is early if its completion time is shorter than its due date. On the contrary, a job is tardy if its completion time is larger than its due date. When the completion time is equal to the due date, the job is on time. The goal of the problem is to arrange the queue's jobs to minimize a specific objective function. In this work, we will consider the minimization of two different objective functions: total tardiness ($\Gamma 1$) and the total earliness and tardiness ($\Gamma 2$) of the jobs. The two objectives are calculated as:

- $\Gamma 1 = \sum_{j \in \mathcal{J}} T_j$,
- $\Gamma 2 = \sum_{j \in \mathcal{J}} (E_j + T_j)$,

where T_j and E_j represent the tardiness and the earliness, respectively, and are computed as $T_j := \max\{0, c_j - d_j\}$ and $E_j := \max\{0, d_j - c_j\}$. They are among the most widely used objectives in scheduling, focusing on meeting jobs due dates. In particular, the minimization of the second objective characterizes the Just-In-Time principle in production.

The motivation to study a single-machine problem relies on the fact that, in plastic and rubber manufacturing, transforming raw material into a final product goes through one or two machines. On the other hand, even those manufacturing require multiple-machine scheduling problems. Each machine represents a chain's primary block. Thus improper usage of a machine can slow down the whole production process.

The easiest way to deal with scheduling in a dynamic context is by using the so-called *dispatching rules*. These rules prioritize jobs waiting for being processed and then select the job with a greedy evaluation whenever a machine gets free (see Sect. 2 for more details). While most dispatching rules schedule on a local view basis, other smarter approaches can provide better results in the long run. For instance, Reinforcement Learning (RL) is a continuing and goal-directed learning paradigm,

[1] Plastic and Rubber 4.0. Piattaforma Tecnologica per la Fabbrica Intelligente (Technological Platform for Smart Factory), URL: https://www.openplast.it/en/homepage-en/.

and it represents a promising approach to deal with online scheduling. The potential of RL on online scheduling has been revealed in several works (see, e.g., [14, 31, 39]). However, while most works compare a single RL algorithm with commonly-used dispatching rules, they do not compare different RL algorithms. A research question naturally arises: how do different RL algorithms perform on online scheduling?

Motivated by investigating the applicability of RL algorithms on online single-machine scheduling in detail, in this work, we will compare the following approaches' performance:

- a random assignment (*Random*) which simply selects a job randomly;
- one of the most popular dispatching rules, namely the *earliest due date* (*EDD*) rule;
- four RL approaches, namely *Q-learning*, *Sarsa*, *Watkins's Q(λ)*, and *Sarsa(λ)*.

Furthermore, we will test the algorithms under different operating conditions (e.g., the frequency of job arrivals).

Therefore, we contribute the literature on two different aspects: getting insights on the compared methods and giving practitioners suggestions on selecting the best method against the specific situation. Notice that comparing and evaluating different algorithms against various aspects and performance indicators is a commonly adopted research methodology (see, e.g., [4, 7–11, 15]). The specific comparison of RL algorithms can be found, for instance, in the game field. In [35], the authors compared two RL algorithms (*Q-learning* and *Sarsa*) through the simulation of bargaining games. Even though the two algorithms present slight differences, they might have essentially different simulation results, as reflected in our experiment (see Sect. 4).

Finally, we also propose some preliminary results obtained by the use of *Deep Q Network* (*DQN*), which utilizes the power of neural networks to approximate the value function (see [25] for a review about DQN). However, our experiments will show that *DQN* is better suited for high-dimensional inputs. In contrast, with smaller input settings, *DQN* has a longer training time and obtains results that are far from the performance of *Watkins's Q(λ)*.

The rest of the paper is organized as follows. Section 2 is dedicated to a general overview of RL techniques, while Sect. 3 introduces and reviews some previous works using RL approaches on scheduling problems. Section 4 describes the algorithmic framework for the online single-machine problem. Section 5 defines the simulation procedure, and the simulation results from three different types of experiments (Sect. 6). Finally, in Sect. 7, the paper concludes with a summary of the findings and some future lines.

2 Reinforcement Learning

RL is a branch of Machine Learning that improves automatically through experience. It comes from three main research branches: the first relates to learning by trial-and-

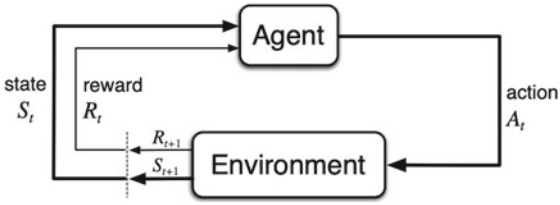


Fig. 1 The agent-environment interaction in RL [33]

error, the second relates to optimal control problems, and the last links to temporal-difference methods (see [33]). The three approaches converged together in the late eighties to produce the modern RL.

RL approaches can be applied to scenarios in which a decision-maker called *agent* interacts with a set of *states* called *environment* by means of a set of possible *actions*. A *reward* is given to the agent in each specific state. In this paper, we consider a discrete time system, i.e. defined over a finite set $\mathcal{T}$ of time steps with its cardinality being called *time horizon*. As shown in Fig. 1, at each time step $t \in \mathcal{T}$, an agent in state S_t takes action A_t, then, the environment reacts by changing into state S_{t+1} and by rewarding the agent of R_{t+1}. The interaction starts from an initial state, and it continues until the end of the time horizon. Such a sequence of actions is named an *episode*. In the following, $\mathcal{E}$ will represent the set of episodes.

Each *state* of the system is associated with a *value function* that estimates the expected future reward achievable from that state. Each state-action pair (S_t, A_t) is associated with a so-called *Q-function* $Q(S_t, A_t)$ that measures the future reward achievable by implementing action A_t in state S_t. The agent's goal is to find the best *policy*, which is a function mapping the set of states to the set of actions, maximizing the cumulative *reward*. If exact knowledge of the Q-function is available, the best policy for each state is defined by $\max_a Q(S_t, a)$.

To estimate the value functions $Q(s, a)$ and discover the optimal policies, three main classes of RL techniques exist Monte Carlo (MC)-based, Dynamic Programming (DP)-based methods, and temporal-difference (TD)-based methods. Unlike DP-based methods, which require complete knowledge of all the possible transitions, MC-based techniques only require some experience and the possibility to sample from the environment randomly. TD-based methods are a sort of combination of MC-based and DP-based ones: they sample from the environment like in MC-based methods and perform updates based on current estimates like DP-based ones. Moreover, TD-based techniques are also appreciated for being flexible, easy to implement, and computationally fast. For these reasons, in this paper, we will consider only RL algorithms belonging to the TD-based methods. Even if several TD-based RL algorithms have been introduced in the literature, the most used are *Sarsa* (an acronym for State-Action-Reward-State-Action), *Q-learning* and their variations, e.g. the *Watkins's Q(λ)* method and the *Sarsa(λ)* (see [36]).

3 Literature Review

Since the first research on scheduling problem was performed in the mid-1950s, many articles have been published in the literature, considering different problem variants and solution approaches.

The manufacturing industries sometimes include a machine bottleneck, which affects, in some cases, all the jobs. Studies on single machine scheduling problems have been gaining importance for a long time since this bottleneck's management is crucial. The excellent surveys by Pinedo [28], Adamu and Adewumi [1], and the work proposed by Leksakul and Techanitisawad [21] have detailed the literature on the theory and applications about this problem in the past several decades.

In the manufacturing environment, various objectives can be considered to use the resources and provide good customer service efficiently. Scheduling against due dates has received considerable attention to meet principles like Lean Management, Just-in-Time, Simultaneous Engineering, etc. For example, the Just-in-Time principle states that jobs are expected to be on time since both late and early processing may negatively influence the manufacturing costs. While late processing does not meet customer expectations, early processing increases inventory costs and causes possible wastes since some products have a limited lifetime. One of the pioneers addressing minimizing the sum of earliness and tardiness (also referred to as the sum of deviations from a common due date) was [19]. Ying [38] addressed a single-machine problem against common due dates concerning earliness and tardiness penalties. He proposed a recovering beam search algorithm to solve this problem. Behnamian et al. [2] considered the problem of parallel machine scheduling to minimize both makespan and total earliness and tardiness. Fernandez-Viagas et al. [12] studied the problem of scheduling jobs in a permutation flow shop to minimize the sum of total tardiness and earliness. They developed and compared four heuristics to deal with the problem. More recently, the two-machine permutation flow shop scheduling problem to minimize total earliness and tardiness has been addressed by two branch-and-bound algorithms utilizing lower bounds and dominance conditions [30].

Total tardiness minimization is another common criterion in the scheduling literature where only the tardiness penalties are considered. Koulamas [20] surveyed theoretical developments, exact and approximation algorithms for the single-machine scheduling problem with the aim of total tardiness minimization. In [26], single machine scheduling with family setup and resource constraints to minimize total tardiness minimization was addressed. A mathematical formulation and a heuristic solution approach were presented. Recently, Silva et al. [24] studied the single machine scheduling problem that minimizes the total tardiness. They presented two algorithms to deal with the situation in which the processing time is uncertain.

As for the scheduling modes, research on online scheduling is one of the popular streams. Since this problem has been an active field for several decades, an in-depth analysis of the literature review is beyond the present paper's scope. Thus, in this section, we recall some of the most traditional approaches to online scheduling, and we review the main applications of RL to this problem.

Differently from tailored algorithms (heuristic and exact methods), which might require effort in implementation and calibration over a broad set of parameters, dispatching rules are widely adopted for online scheduling for their simplicity (see, e.g., [18]). For instance, the *earliest due date* (*EDD*) dispatching rule is one of the most commonly used ones in practical applications [34]. *EDD* schedules first the job with the earliest due date. Again, in [16], the authors propose a deterministic greedy algorithm known as *list scheduling* (*LS*), which assigns each job to the machine with the smallest load. For more details, we refer the reader to the work [27] that classified over one hundred dispatching rules. In [6], the authors designed a deterministic algorithm and a randomized one for online machine sequencing problems using Linear Programming techniques. At the same time, in [23], the authors proposed an algorithm to make jobs artificially available to the online scheduler by delaying the release time of jobs.

In online scheduling, a decision-maker is regularly scheduling jobs over time, attempting to reach the overall best performance. Therefore, it is reasonable that RL represents one of the possible techniques to exploit such a setting.

In [14], the authors interpreted job-shop scheduling problems as sequential decision processes. They try to improve the job dispatching decisions of the agent by employing an RL algorithm. Experimental results on numerous benchmark instances showed the competitiveness of the RL algorithm. More recently, in [39], the authors modeled the scheduling problem as a Markov Decision Process and solved it through a simulation-based value iteration and a simulation-based *Q-learning*. Their results clearly showed that such RL algorithms could achieve better performance concerning several dispatching heuristics, disclosing RL application's potential in the field. In the context of an online single-machine environment, in [37], the authors compared the performance of *neural fitted Q-learning* techniques using combinations of different states, actions, and rewards. They proved that taking only the necessary inputs of states and actions is more efficient.

While all the discussed works revealed RL's competitiveness on scheduling problems, a further comparison of the performance among various RL algorithms is still missing in the scheduling literature. With the knowledge of the available studies showing RL's potential and the demand from the industrial application, we are motivated to compare different RL approaches' performance on online scheduling for getting more insights. In particular, we carry out experimental studies on four of the most commonly used model-free RL algorithms, namely *Q-learning*, *Sarsa*, *Watkins's Q(λ)*, and *Sarsa(λ)*. Our comparison methodology is inspired by [37], in which the best configuration for minimizing maximal lateness is pursued. In our work, instead, we propose two different objective functions to minimize: the total tardiness and the total earliness and tardiness. Moreover, another significant difference with their work lies in the way we evaluate the results. While they used the result from one run, our results come from 50 runs with different random seeds, and two different time step sizes are tested (the interaction between agent and environment is checked in each step). We further test a neural network-based RL technique showing that it is unnecessary to use such a combination when the state space is limited.

4 Reinforcement Learning Algorithms for Online Scheduling

In this section, we describe the algorithmic framework used to deal with our online single-machine scheduling problem. In particular, we provide several variants based on different RL techniques.

4.1 States, Actions, and Rewards

To be approached by RL techniques, we define our problem setting along the lines used in [37]. In particular:

- *state*: a state is associated with each possible length of the jobs in the waiting queue;
- *action*: if not all the jobs are finished, the action is either to select one new job from a specific position of the waiting queue and start processing it (we recall that preemption is allowed) or to continue processing the job which has been already assigned to the machine in the previous step;
- *reward*: since RL techniques aim at maximizing rewards while our problem seeks to minimize its objective function (either the total tardiness or the total earliness and tardiness), we set the reward of a state as the opposite value of the considered measure.

When the action implies selecting a job from a certain position in the waiting queue, it is important to decide the order in which jobs are stored inside the queue. Therefore, we implemented three possible ordering of jobs that provide very different scheduling effects:

- jobs are unsorted (*UNSORT*), i.e., they have the same order as the arrivals;
- jobs are sorted by increasing value of due time (*DT*);
- all unfinished jobs are sorted by increasing the value of the sum of due time and processing time (*DT*+*PT*).

For instance, by using *DT*, if the action is to select a job in the second position of the queue, the job with the second earliest due time will be processed.

4.2 RL Algorithms Adopted

We have decided to implement four different RL algorithms, namely *Q-learning*, *Sarsa*, *Watkins's Q(λ)*, and *Sarsa(λ)*. They are described in the following. Here are some notations used:

- s state;

Fig. 2 An example of Q table

Q Table		Actions				
		Select Job 1	Select Job 2	Select Job 3	Select Job 4	Select Job 5
States	0	0	0	0	0	0
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.
	5	-20	-15	-34	-14	-31
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.
	10	-15	-21	-22	-16	-23

- a action;
- $\mathcal{S}$ set of nonterminal states;
- $\mathcal{A}(s)$ set of actions possible in state s;
- S_t state at t;
- A_t action at t;
- R_t reward at t.

4.2.1 Q-Learning

Q-learning is a technique that learns the value of an optimal policy independently of the agent's action. It is largely adopted for its simplicity in the analysis of the algorithm and for the possibility of early convergence proofs by directly approximating the optimal action-value function (see [33, 36]). The updating rule for the estimation of the Q-function is:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)]. \quad (1)$$

The $Q(S_t, A_t)$ function estimates the quality of state-action pair. At each time step t, the reward R_{t+1} from state S_t to S_{t+1} is calculated and $Q(S_t, A_t)$ is updated accordingly. The coefficient α is the learning rate ($0 \leq \alpha \leq 1$); it determines the extent that new information overrides the old information. Furthermore, γ is the discount factor determining the importance of future reward and finally, $\max_a Q(S_{t+1}, a)$ is the estimation of best future value.

The values of the Q-function are stored in a look-up table called *Q-table*. Figure 2 displays an example of Q-table storing Q-function values for states from 0 to 10 (in row) and actions from selecting *Job 1* to *Job 5* (in column). By overlooking the actual policy being followed in deciding the next action, *Q-learning* simplifies the analysis of the algorithm and enabled early convergence proofs.

4.2.2 Sarsa

Sarsa is a technique that updates the estimated Q-function by following the experience gained from executing some policies (see [32, 33]). The updating rule for the estimation of the Q-function is:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]. \tag{2}$$

The structure of formula (2) is similar to (1). The only difference is that (2) considers the actual action implemented in the next step A_{t+1}, instead of the generic best action $\max_a Q(S_{t+1}, a)$.

As for *Q-learning*, also in *Sarsa* the values of the Q-function are stored in a *Q table*. Despite the more expensive behaviour with respect to *Q-learning*, *Sarsa* may provide better online performances in some scenarios (as shown by the *Cliff Walking* example in [33]).

4.2.3 Watkins's Q(λ)

Watkins's Q(λ) is a well-known variant of *Q-learning*. The main difference with respect to classical *Q-learning* is the presence of a so-called *eligibility trace*, i.e. a temporary record of the occurrence of an event, such as the visiting of a state or the taking of an action. The trace marks the memory parameters associated with the event as eligible for undergoing learning changes. A trace is initialized when a state is visited or an action is taken, and then the trace gets decayed over time according to a decaying parameter λ (with $0 \leq \lambda \leq 1$). Let us call $e_t(s, a)$ the trace for a state-action pair (s, a). Let us also define an indicator parameter $\mathbb{1}_{xy}$ that takes value 1 if and only if x and y are the same, and 0 otherwise. Then, for any (s, a) pair (for all $s \in \mathcal{S}$, $a \in \mathcal{A}$), the updating rule for the estimation of the Q-function is:

$$Q_{t+1}(s, a) = Q_t(s, a) + \alpha \delta_t e_t(s, a) \tag{3}$$

where

$$\delta_t = R_{t+1} + \gamma \max_{a'} Q_t(S_{t+1}, a') - Q_t(S_t, A_t) \tag{4}$$

and

$$e_t(s, a) = \gamma \lambda e_{t-1}(s, a) + \mathbb{1}_{sS_t} \mathbb{1}_{aA_t} \tag{5}$$

if $Q_{t-1}(S_t, A_t) = \max_a Q_{t-1}(S_t, a)$, and $\mathbb{1}_{sS_t} \mathbb{1}_{aA_t}$ otherwise.

As the reader can notice, by plugging Eq. (4) into Eq. (3), we obtain an equation similar to (1) but with the additional eligibility term that increments the value of δ_t if the state and action selected by the algorithm are one of the eligibility states. In the rest of the paper we use *Q(λ)* referring to *Watkins's Q(λ)*.

4.2.4 Sarsa(λ)

Similarly to *Q(λ)*, the *Sarsa(λ)* algorithm represents a combination between *Sarsa* and eligibility traces to obtain a more general method that may learn more efficiently. Here, for any (s, a) pair (for all $s \in \mathcal{S}$, $a \in \mathcal{A}$), the updating rule for the estimation of the Q-function is:

$$Q_{t+1}(s, a) = Q_t(s, a) + \alpha \delta_t e_t(s, a) \tag{6}$$

where

$$\delta_t = R_{t+1} + \gamma Q_t(S_{t+1}, A_{t+1}) - Q_t(S_t, A_t) \tag{7}$$

and

$$e_t(s, a) = \gamma \lambda e_{t-1}(s, a) + \mathbb{1}_{sS_t} \mathbb{1}_{aA_t} \tag{8}$$

Unlike Eq. (5), there is no other condition (set the eligibility traces to 0 whenever a non-greedy action is taken) added. A more in-depth discussion about the interpretation of the formulas is given in [33].

5 Simulation Setting

In order to perform the comparison under interest, we create an online scheduling simulation procedure as described in Algorithm 1.

Algorithm 1 Online scheduling simulation through RL algorithms

Require: $|\mathcal{E}|$ number of episodes; $|\mathcal{T}|$ number of time-steps;
Initialize $Q(s, a) = 0, \forall\, s \in \mathcal{S}, a \in \mathcal{A}$;
for $\eta \leftarrow 1$ to $|\mathcal{E}|$ **do**
  Initialize S
  for $t \leftarrow 1$ to $|\mathcal{T}|$ **do**
    if new jobs arrive **then**
      Update waiting list L
    end if
    if L is not empty **then**
      Take A_t in S_t, observe R_t, S_{t+1}
      Calculate A_{t+1} and update Q_t
      $S_t \leftarrow S_{t+1}$, $A_t \leftarrow A_{t+1}$
    end if
  end for
end for

We first update Q tables through a training phase then use the Q tables to select actions in the test phase.

The arrival time of job j are distributed according to an exponential distribution, i.e., $X_j \sim exp(r)$ with the rate parameter valued $r = 0.1$. It is simulated in this way: at the first time step, a random number of jobs (from 1 to 6 jobs) and an interval time (following the exponential distribution) are generated. Once a job is generated (simulating the job's arrival), it will immediately be put into the waiting queue. Then at the next time step, if the interval time is passed, new jobs will be generated and put into the waiting queue; meanwhile, a new interval time will be created. Otherwise, nothing is created. Then the same procedure repeats till reaching a final state.

For the settings regarding RL algorithms:

- In the policy, $\epsilon = 0.1$ enabling highly greedy actions while keeping some randomness in job selections;
- In the value function, $\alpha = 0.6$, i.e., there is a bit higher tendency to explore more possibilities while a bit lower in keeping exploiting old information, whereas $\gamma = 1.0$, which means it strives for a long-term high reward;
- In the eligibility traces, λ is 0.95, a high decaying value leads to a longer-lasting trace.

It is worth noting that all the algorithms considered are heuristics. They focus on finding a good solution quickly by finding a balance between the solution space's intensified and diversified explorations. Nevertheless, the direct implantation of the algorithms above does not ensure enough diversification. For this reason, it is common to use a ϵ-greedy method. Thus, with probability ϵ, exploration is chosen, which means the action is chosen uniformly at random between the available ones. Instead, with probability $1 - \epsilon$, exploitation is chosen by taking the actions with the highest values greedily. After knowing how to balance exploration and exploitation, we need to define a learning method for finding out policies leading to higher cumulative rewards.

In an episode, we start a new schedule by initializing state S and terminates when either reaching the maximum steps or no jobs to process. To simulate real-time scheduling, for each episode, we check the arrivals of new jobs and update the waiting queue if there are, then we choose the action A, and calculate the reward R and the next state S' accordingly. The Q-functions are updated according to the exact RL algorithms used. The same procedure is carried out in both training and test phases except that in the test. The Q-table is not initialized with empty values but obtained from the training phase.

Let us show how the total tardiness value evolves, for an example in which *Q-learning* is used to schedule the jobs. In Fig. 3, the graph on the bottom shows that the reward increases and reaches the maximum and holds steady after 80 episodes. Accordingly, the objective value (the total tardiness) decreases with more noticeable fluctuations and drops more slowly after 80 episodes. While the reward keeps stable, total tardiness continues dropping to around 40000. To summarize, using total tardiness as a goal is useful, but it is still challenging to represent the trend of this objective value adequately.

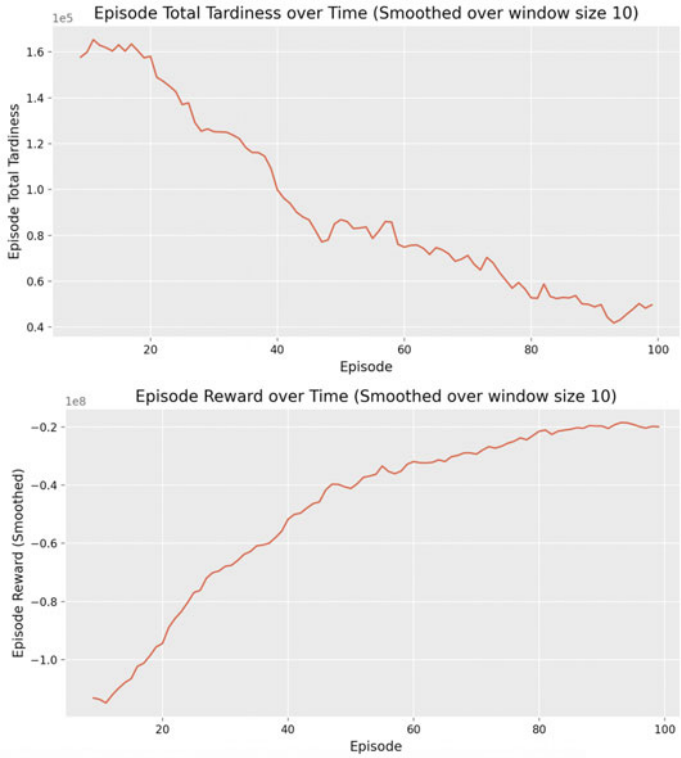


Fig. 3 The changes to reward and the objective value (total tardiness) of 100 episodes

6 Experimental Results and Discussion

In this section, we propose three different experimental results. Section 6.1 compares the performance among random assignment (*Random*), *EDD*, and the four RL approaches implemented for both minimizing the total tardiness and the total earliness and tardiness. Section 6.2 investigates the possible impact of different operating conditions (i.e., frequency of jobs arrivals) on the RL approaches. Finally, Sect. 6.3 compares *Q(λ)* and *DQN*.

The algorithms have been implemented in Python 3.6. To avoid possible ambiguities, we locate the related code in a public repository.[2] All the experiments are carried out on an *Intel Core i5* CPU@2.3 GHz machine equipped with 8GB RAM and running *MacOS* v10.15.4 operating system.

[2] https://github.com/Yuanyuan517/RL_OnlineScheduling.git.

Table 1 Simulations of the algorithms with different settings and considering the total tardiness minimization

Algorithm	Jobs order	$\lvert\mathcal{T}\rvert = 2500$		$\lvert\mathcal{T}\rvert = 5000$	
		avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)	avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)
Random	–	2.59	0.50	3.06	0.69
EDD	–	7.67	1.76	9.19	1.47
Q-learning	*UNSORT*	2.15	0.43	2.04	0.35
Q-learning	*DT*	1.45	0.28	1.29	0.20
Q-learning	*DT+PT*	1.44	0.30	1.25	0.18
Sarsa	*UNSORT*	2.55	0.53	2.47	0.39
Sarsa	*DT*	1.65	0.40	1.76	0.36
Sarsa	*DT+PT*	1.66	0.47	1.68	0.33
Sarsa(λ)	*UNSORT*	4.42	0.93	5.04	0.93
Sarsa(λ)	*DT*	7.04	1.35	7.73	1.34
Sarsa(λ)	*DT+PT*	3.08	1.03	7.70	1.33
Q(λ)	*UNSORT*	2.04	0.42	2.01	0.40
Q(λ)	*DT*	**1.11**	**0.18**	1.13	0.17
Q(λ)	*DT+PT*	1.19	0.26	**1.09**	**0.14**

6.1 RL Algorithms Versus Random and EDD

To check if considering different time horizons leads to different results, we consider two experiments in which the time horizon $\mathcal{T}$ is set to 2500 and 5000, respectively. For each of the settings, we ran 50 tests with different random seeds. For each algorithm Θ, we call $\Gamma_{\zeta\Theta}$ the objective value achieved in simulation ζ. Furthermore, we define $\rho_{\zeta\Theta}$ to be the percentage gap between the objective value achieved by the best algorithm and algorithm Θ during run ζ, i.e.,

$$\rho_{\zeta\Theta} = \frac{\Gamma_{\zeta\Theta}}{\min_{\zeta\Theta} \Gamma_{\zeta\Theta}}. \tag{9}$$

To compare the different algorithms, we consider the average value of $\rho_{\zeta\Theta}$ concerning all the runs.

The simulation results with the algorithms (under different job orders, time horizons) for the total tardiness and the total earliness and tardiness minimization are displayed in Tables 1 and 2, respectively. Note that avg($\rho_{\zeta\Theta}$) and std($\rho_{\zeta\Theta}$) represent respectively the mean value and standard deviations of $\rho_{\zeta\Theta}$. The best value among all the combinations of algorithms and jobs order policies for each time horizon is highlighted in bold font.

While in [37] the authors show that *EDD* gets a better result than RL in minimizing the maximum tardiness, as shown in Table 1, all the implemented RL algorithms outperform *EDD* in minimizing the total tardiness. This result is exciting and proba-

Table 2 Simulations of the algorithms with different settings and considering the total earliness and tardiness minimization

Algorithm	Jobs order	$\lvert\mathcal{T}\rvert = 2500$		$\lvert\mathcal{T}\rvert = 5000$	
		avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)	avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)
Random	–	5.85	9.95	19.34	42.96
EDD	–	4.17	**1.86**	**6.24**	**2.99**
Q-learning	*UNSORT*	5.33	9.33	12.62	30.34
Q-learning	*DT*	3.95	6.71	10.20	23.10
Q-learning	*DT+PT*	3.72	6.20	9.91	22.20
Sarsa	*UNSORT*	5.72	9.62	17.45	39.43
Sarsa	*DT*	4.43	7.87	16.34	37.85
Sarsa	*DT+PT*	4.46	8.13	13.77	33.89
Sarsa(λ)	*UNSORT*	10.77	17.78	36.59	75.93
Sarsa(λ)	*DT*	13.29	23.84	49.71	111.14
Sarsa(λ)	*DT+PT*	6.04	9.87	55.77	116.36
Q(λ)	*UNSORT*	4.68	8.25	15.45	34.97
Q(λ)	*DT*	3.89	6.66	10.21	23.98
Q(λ)	*DT+PT*	**3.29**	5.62	9.23	21.36

bly depends on whether the learning paradigm is more tailored to optimize min-sum problems than min-max ones. Also, it can be seen that the size of running time steps influences the result on job order but does not affect the algorithm. For the case with 2500 steps, the configuration *Q(λ)* plus *DT* gets the best result, instead for 5000 steps, the configuration *Q(λ)* plus *DT+PT* outperforms the others.

Besides, we find with the sorting choice *DT+PT* that all algorithms get smaller average values except for the configuration *Q(λ)* with 2500 steps. Comparatively, a randomly sorting job leads to a much worse result.

Instead, as reported in Table 2, *EDD* outperforms the other algorithms in minimizing the total earliness and tardiness, both in terms of both the mean and the standard deviation for the larger time horizon. Moreover, it achieves the smallest standard deviation for both time horizons. However, the configuration using *Q(λ)* and *DT+PT* gets the smallest mean for the case with 2500 time steps. Taking into account the three job's ordering, it can be noticed that all the algorithms in combination with *UNSORT* have the worst results in terms of both the mean and the standard deviation, except for the algorithm *Sarsa(λ)* (which instead performs very poorly with the sorting choice *DT*).

Finally, it can be noticed that the mean and the standard deviation obtained by the algorithms in minimizing the total earliness and tardiness are larger than those achieved in Table 1. Unlike the total tardiness minimization's objective, the total earliness and tardiness may not be well addressed by the proposed RL algorithms. Considering the measure of jobs, earliness can negatively affect the effectiveness of RL algorithms. A possible reason can be found in the test environment settings. In

Table 3 Experiments on the rate parameter with best settings from *Q(λ)* concerning the total tardiness minimization

Jobs order	r	avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)
DT	0.05	**1.14**	**0.18**
DT+PT	0.05	1.17	0.55
DT	0.10	**1.10**	**0.17**
DT+PT	0.10	1.17	0.26
DT	0.20	1.17	0.28
DT+PT	0.20	**1.12**	**0.24**

the experiments, the due date is calculated by first taking a random value, namely the processing time of the job, from an exponential distribution $X \sim Exp(\iota)$ where

$$\iota = \frac{1}{7 \times \max_{j \in \mathcal{J}}\{processingTimeJob_j\}},$$

and adding that value to the current simulation time. Reminding that the tardiness of a job j is defined as $T_j := \max\{0, c_j - d_j\}$ where $c_j = startTime_j + processingTimeJob_j$, then the more the jobs accumulated as time running, the bigger the difference between the start time and due date for a job. Hence, more delays will occur, which might cause the simulation results in favor of tardiness calculation.

6.2 *Q(λ) Performance Against Different Job Arrival Rates*

We carried out another test against different frequencies of job arrivals (controlled by the rate parameter r) by considering the two best RL algorithm combinations resulted from the previous tests, i.e., *Q(λ)* plus *DT* and *Q(λ)* plus *DT+PT*. To understand whether the value of r affects the performance, we experimented with 2 more values, i.e. $r = \{0.05, 0.2\}$ in addition to the previous one $r = 0.1$. Tables 3 and 4 show the results of this test in the case of minimization of total tardiness and total earliness and tardiness, respectively. Note that the results have been normalized by following Eq. (9) with 50 tests and $|\mathcal{T}| = 2500$ for each test.

As shown in the Table 3, with small values of r (e.g., 0.05, 0.10), i.e., when jobs arrive much less frequently than the last one, the version with jobs ordered by *DT* performs better. When jobs arrive much more frequently, the version sorted by *DT+PT* wins. Hence, a careful selection of algorithms and settings according to the operating conditions matters.

Table 4 shows results on comparing the total earliness and tardiness with the same settings as the ones of Table 3. However, even with a different objective, the results

Table 4 Experiments on the rate parameter with best settings from *Q(λ)* concerning the total earliness and tardiness minimization

Jobs order	r	avg($\rho_{\zeta\Theta}$)	std($\rho_{\zeta\Theta}$)
DT	0.05	**1.74**	**0.83**
DT+PT	0.05	1.94	0.90
DT	0.10	3.89	6.66
DT+PT	0.10	**3.29**	**5.62**
DT	0.20	5.97	6.66
DT+PT	0.20	**5.75**	**6.37**

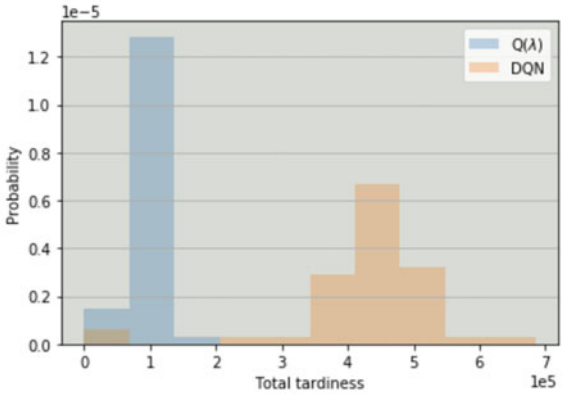


Fig. 4 Comparison between *Q(λ)* and DQN on the total tardiness of 50 runs with different seeds representing different schedules

for $r = 0.05$ and $r = 0.20$ are similar: the version using DT (for the former) and $DT + PT$ (for the latter) perform better. The difference lies on $r = 0.10$, which gets better performance with $DT + PT$ instead of DT in Table 3. Thus, a combination of factors (settings, operating conditions, and objective) is clearly necessary to be considered when selecting the RL algorithm.

6.3 Comparison Between Q(λ) and DQN

Finally, in this section, we compare a four-layer DQN and *Q(λ)* plus *DT+PT*, which is the best performing RL algorithm. Figure 4 shows such a comparison in the total tardiness minimization, while Fig. 5 is dedicated to the case minimizing the total earliness and tardiness. We run 50 tests and $|\mathcal{T}| = 5000$ in each test. The horizontal axis represents the total tardiness and the vertical axis shows the probability the objective value falls in. Note that the brown area indicates the overlapping between *Q(λ)* and DQN.

From Fig. 4, we can see *Q(λ)* has much higher probability with smaller objective value, which indicates *Q(λ)* outperforms DQN. Taking into account the time spent

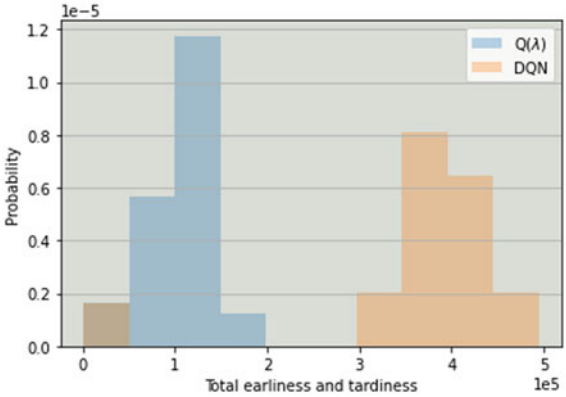


Fig. 5 Comparison between *Q(λ)* and DQN on the total earliness and tardiness of 50 runs with different seeds representing different schedules

in training DQN is almost 10 times of *Q(λ)*, *Q(λ)* is a better option, especially for guaranteeing a flexible and adaptive scheduling in realtime.

The results in Fig. 5 are very similar to the previous ones. Compared to *Q(λ)*, DQN has a much higher probability with a bigger objective, which stands for its poor performance.

7 Conclusions and Future Research Directions

In this paper, we compared four RL methods, namely *Q-learning*, *Sarsa*, *Watkins's Q(λ)*, and *Sarsa(λ)*, with *EDD* and random assignment on an online single-machine scheduling problem with two different objectives, as the total tardiness and the total earliness and tardiness minimization. The experiments show that:

- better scheduling performance in minimizing the total tardiness is achieved by the RL method *Watkins's Q(λ)*, especially when the action concerns the selection of jobs sorted by due date for the smaller time horizon ($|\mathcal{T}| = 2500$) and the selection of jobs sorted by due date and processing time for bigger time horizon ($|\mathcal{T}| = 5000$).
- considering the measure of earliness may negatively affect the performance of RL algorithms. In minimizing the total earliness and tardiness, *Watkins's Q(λ)* with the sorting choice *DT+PT* performs better for the small-time horizon in terms of mean values. In contrast, *EDD* can get better results for the large-time horizon.
- when considering different frequencies of jobs arrival, the combination of *Q(λ)* and job orders have different performances in various operating conditions with different objectives.
- slight differences in algorithms and objectives can profoundly change the results.

Besides, with limited input, using DQN is too costly for extended running time and energy spent adjusting parameters to guarantee a good result. In addition to the

numerical results explicitly presented in the paper, according to our previous experience, RL algorithms also do not perform well on a single job-related objective (e.g., maximum tardiness [37]). These indicate careful analysis should be done from different viewpoints (running time, operating conditions, average results from multiple experiments) for making a wiser selection of algorithms.

Furthermore, with multiple machines, more transitions must be considered, which need more representational state information. Thus it will be impossible to store values of all state-action pairs in a Q-table. DQN may take a leading role then. As indicated by the work [13], unpredictable changes may happen at different places in the state-action space, and more care should be taken to avoid instabilities of DQN. One technique that can achieve this goal is the so-called *kernel function* (see [5]), which builds a future research avenue. Another possibility is creating an algorithm selection framework, as explored in work by Rice [29]. In particular, by mapping from the problem characteristics to the appropriate algorithms considered in the framework, we can achieve an automatic selection of the best one to use.

Acknowledgements This research was partially supported by the Plastic and Rubber 4.0 (P&R4.0) research project, POR FESR 2014–2020 - Action I.1b.2.2, funded by Piedmont Region (Italy), Contract No. 319-31. The authors acknowledge all the project partners for their contribution.

References

1. Adamu, M.O., Adewumi, A.: A survey of single machine scheduling to minimize weighted number of tardy jobs. J. Ind. Manag. Optim. **10**, 219–241 (2014)
2. Behnamiana, J., Ghomi, S.F., Zandieh, M.: A multi-phase covering pareto-optimal front method to multi-objective scheduling in a realistic hybrid flowshop using a hybrid metaheuristic. Expert Syst. Appl. **36**, 11057–11069 (2009)
3. Brucker, P.: Scheduling Algorithms, 5th edn. Springer Publishing Company, Incorporated (2010)
4. Castrogiovanni, P., Fadda, E., Perboli, G., Rizzo, A.: Smartphone data classification technique for detecting the usage of public or private transportation modes. IEEE Access **8**, 58377–58391 (2020). https://doi.org/10.1109/ACCESS.2020.2982218
5. Cerone, V., Fadda, E., Regruto, D.: A robust optimization approach to kernel-based nonparametric error-in-variables identification in the presence of bounded noise. In: 2017 American Control Conference (ACC), IEEE (2017). https://doi.org/10.23919/ACC.2017.7963056
6. Correa, J.R., Wagner, M.R.: Lp-based online scheduling: from single to parallel machines. Math. Program. **119**(1), 109–136 (2009)
7. Fadda, E., Plebani, P., Vitali, M.: Optimizing monitorability of multi-cloud applications. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) Advanced Information Systems Engineering. CAiSE 2016. Lecture Notes in Computer Science, vol. 9694, pp. 411–426. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_25
8. Fadda, E., Perboli, G., Squillero, G.: Adaptive batteries exploiting on-line steady-state evolution strategy. In: Squillero, G., Sim, K. (eds.) Applications of Evolutionary Computation. EvoApplications 2017. Lecture Notes in Computer Science, vol. 10199, pp. 329–341. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55849-3_22
9. Fadda, E., Manerba, D., Tadei, R., Camurati, P., Cabodi, G.: KPIs for optimal location of charging stations for electric vehicles: the Biella case-study. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.) Proceedings of the 2019 Federated Conference on Computer Science and

Information Systems, IEEE, Annals of Computer Science and Information Systems, vol. 18, pp. 123–126 (2019). https://doi.org/10.15439/2019F171
10. Fadda, E., Manerba, D., Cabodi, G., Camurati, P., Tadei, R.: Evaluation of Optimal Charging Station Location for Electric Vehicles: An Italian Case-Study, pp. 71–87 (2021). https://doi.org/10.1007/978-3-030-58884-7_4
11. Fadda, E., Manerba, D., Cabodi, G., Camurati, P.E., Tadei, R.: Comparative analysis of models and performance indicators for optimal service facility location. Transp. Res. Part E: Logist. Transp. Rev. **145** (2021)
12. Fernandez-Viagas, V., Dios, M., Framinan, J.M.: Ecient constructive and composite heuristics for the permutation flowshop to minimise total earliness and tardiness. Comput. Oper. Res. **75**, 38–48 (2016)
13. François-Lavet, V., Fonteneau, R., Ernst, D.: How to discount deep reinforcement learning: towards new dynamic strategies (2015). arXiv:151202011
14. Gabel, T., Riedmiller, M.: Adaptive reactive job-shop scheduling with reinforcement learning agents. Int. J. Inf. Technol. Intell. Comput. **24**(4), 14–18 (2008)
15. Giusti, R., Iorfida, C., Li, Y., Manerba, D., Musso, S., Perboli, G., Tadei, R., Yuan, S.: Sustainable and de-stressed international supply-chains through the synchro-net approach. Sustainability **11**, 1083 (2019). https://doi.org/10.3390/su11041083
16. Graham, R.L.: Bounds for certain multiprocessing anomalies. Bell Syst. Tech. J. **45**(9), 1563–1581 (1966). https://doi.org/10.1002/j.1538-7305.1966.tb01709.x
17. Graves, S.C.: A review of production scheduling. Oper. Res. **29**(4), 646–675 (1981). https://doi.org/10.1287/opre.29.4.646
18. Kaban, A., Othman, Z., Rohmah, D.: Comparison of dispatching rules in job-shop scheduling problem using simulation: a case study. Int. J. Simul. Model. **11**(3), 129–140 (2012). https://doi.org/10.2507/IJSIMM11(3)2.201
19. Kanet, J.: Minimizing the average deviation of job completion times about a common due date. Nav. Res. Logist. Q. **28**, 643–651 (1981)
20. Koulamas, C.: The single-machine total tardiness scheduling problem: review and extensions. Eur. J. Oper. Res. **202**, 1–7 (2010)
21. Leksakul, K., Techanitisawad, A.: An application of the neural network energy function to machine sequencing. Comput. Manag. Sci. **2**, 309–338 (2005)
22. Li, Y., Carabelli, S., Fadda, E., Manerba, D., Tadei, R., Terzo, O.: Machine learning and optimization for production rescheduling in industry 4.0. In: The International Journal of Advanced Manufacturing Technology, pp. 1–19 (2020). https://doi.org/10.1007/s00170-020-05850-5
23. Lu, X., Sitters, R., Stougie, L.: A class of on-line scheduling algorithms to minimize total completion time. Oper. Res. Lett. **31**(3), 232–236 (2003). https://doi.org/10.1016/S0167-6377(03)00016-6
24. Marco Silve, N.M., Poss, Michael: Solution algorithms for minimizing the total tardiness with budgeted processing time uncertainty. Eur. J. Oper. Res. **283**, 70–82 (2020)
25. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning (2013). arXiv:13125602
26. Oliver Herr, G.: Minimising total tardiness for a single machine scheduling problem with family setups and resource constraints. Eur. J. Oper. Res. **248**, 123–135 (2016)
27. Panwalkar, S.S., Iskander, W.: A survey of scheduling rules. Oper. Res. **25**(1), 45–61 (1977). https://doi.org/10.1287/opre.25.1.45
28. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems. Springer, New York, NY, USA (2012)
29. Rice, J.R.: The algorithm selection problem. In: Advances in Computers, vol. 15, pp. 65–118. Elsevier (1976)
30. Schaller, J., Valente, J.: Branch-and-bound algorithms for minimizing total earliness and tardiness in a two-machine permutation flow shop with unforced idle allowed. Comput. Oper. Res. **109**, 1–11 (2019)
31. Sharma, H., Jain, S.: Online learning algorithms for dynamic scheduling problems. In: 2011 Second International Conference on Emerging Applications of Information Technology, pp. 31–34 (2011)

32. Singh, S., Jaakkola, T., Littman, M.L., Szepesvári, C.: Convergence results for single-step on-policy reinforcement-learning algorithms. Mach. Learn. **38**(3), 287–308 (2000). https://doi.org/10.1023/A:1007678930559
33. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT press (2018)
34. Suwa, H., Sandoh, H.: Online Scheduling in Manufacturing: A Cumulative Delay Approach. Springer Science & Business Media (2012)
35. Takadama, K., Fujita, H.: Toward guidelines for modeling learning agents in multiagent-based simulation: implications from q-learning and sarsa agents. In: International Workshop on Multi-Agent Systems and Agent-Based Simulation, pp. 159–172. Springer (2004). https://doi.org/10.1007/978-3-540-32243-6_13
36. Watkins, C.J.C.H.: Learning from delayed rewards. Thesis Submitted for Ph.D., King's College, Cambridge (1989)
37. Xie, S., Zhang, T., Rose, O.: Online single machine scheduling based on simulation and reinforcement learning. In: Simulation in Produktion und Logistik 2019, Simulation in Produktion und Logistik 2019 (2019)
38. Ying, K.C.: Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm. Comput. Ind. Eng. **55**, 494–502 (2008)
39. Zhang, T., Xie, S., Rose, O.: Real-time job shop scheduling based on simulation and markov decision processes. In: 2017 Winter Simulation Conference (WSC), IEEE, pp. 3899–3907 (2017). https://doi.org/10.1109/WSC.2017.8248100

Ant Colony Optimization Algorithm for Fuzzy Transport Modelling: InterCriteria Analysis

Stefka Fidanova, Olympia Roeva, and Maria Ganzha

Abstract Public transport plays an important role in our live. It is very important to have a reliable service. Up to 1000 km, trains and buses play the main role in the public transport. The number of the people and which kind of transport they prefer is important information for transport operators. In this paper is proposed algorithm for transport modeling and passenger flow, based on Ant Colony Optimization method. The problem is described as multi-objective optimization problem. There are two optimization purposes: minimal transportation time and minimal price. Some fuzzy element is included. When the price is in a predefined interval it is considered the same. Similar for the starting traveling time. The aim is to show how many passengers will prefer train and how many will prefer buses according their preferences, the price or the time. The InterCriteria Analysis (ICrA) is applied over numerical results obtained from ACO algorithm in order to estimate the algorithm performance. The ICrA results show that the proposed ACO algorithm performs very well.

Keywords Fuzzy transport modelling · Ant colony optimization · Metaheuristics · Intercriteria analysis · Index matrix · Intuitionistic fuzzy sets

S. Fidanova (✉)
Institute of Information and Communication Technology, Bulgarian Academy of Sciences, Sofia, Bulgaria
e-mail: stefka@parallel.bas.bg

O. Roeva
Institute of Biophysics and Biomedical Engineering, Bulgarian Academy of Sciences, Sofia, Bulgaria
e-mail: olympia@biomed.bas.bg

M. Ganzha
System Research Institute, Polish Academy of Sciences Warsaw and Management Academy, Warsaw, Poland
e-mail: maria.ganzha@ibspan.waw.pl

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_6

1 Introduction

Comfortable transportation from one town to another one is very important. It exists different ways of transportation. The cheaper transport is a railway (excluding the super-fast with velocity more than 200 km/h), but the trains are slower. Buses and fast trains are more expensive, but faster. All this need to be taken in to account, when a transportation model is prepared. In this paper the transportation problem is defined as an optimization problem. It is a multi-objective problem with two objective functions: total time and total price of all passengers. The goal is to minimize the both objective functions. The two objective functions are antithetic, the faster transportation is expensive and the cheaper transportation is slower. Thus when one of the objective functions decreases, the other increases. The problem is multi-objective, therefore is received set of nondominated solutions instead of one optimal solution. The set of solutions is analysed and the final decision, which solution is optimal accordingly with some additional constraints. The solutions of our problem shows how many passenger will use the train and how many will use bus and fast train.

The oldest public transport, among those that are still in use, is the railroads. Nowadays the main concurrencies of the trains are buses, especially in the regions with highways. Thus the models, which can analyse the passenger flow and its preferences, are important for transportation planning. In our model we include some fuzzy element, thus we try to make it more realistic and close to human thinking.

Various transportation models can be found in the literature [12]. The importance of every of the models depends of its functions. One of the models are concentrated on scheduling [1]. Other models are focused on simulation to analyse the level of utilization of different types of transportation [32]. The model in [21] aims to optimize the transportation network design. In [15] is modelled free-way traffic flow. When a network of free-way is given, their model can predict the traffic flow with high accuracy. Our model is focused on modelling the passenger flow according their preferences. The fuzzyfication of the model makes it more realistic, more close to the human thinking. When the price or the time is in some predetermined interval we accept it as the same. The problem shows the distribution of the passenger flow and how it changes when the timetable or type of the vehicles are changed.

The problem is difficult in computational point of view and cannot be solved with traditional numerical methods with reasonable computational resources. It is more appropriate to apply some metaheuristic method on this kind of problems. We apply ant colony optimization algorithm. The model is tested on real problem, the passenger flow between Sofia and Varna, one of the longest destinations in Bulgaria.

In order to evaluate the performance of the considered here ACO algorithm the approach named InterCriteria Analysis (ICrA) is used. ICrA aiming to go beyond the nature of the criteria involved in a process of evaluation of multiple objects against multiple criteria, and, thus to discover some dependencies between the ICrA criteria themselves [2]. The approach is based on the apparatus of the index matrices and the intuitionistic fuzzy sets, two approaches that have been actively researched and applied [8, 22, 28, 30, 31].

For the first time ICrA has been applied for the purposes of temporal, threshold and trends analyses of an economic case-study of European Union member states' competitiveness [9, 10]. The approach already has a lot of different applications [26, 27, 29]. ICrA is applied for comparison of different metaheuristics as GAs and ACO [11, 25], too.

In this paper ICrA is applied for analysis of an ACO algorithm. ACO is applied for transport modelling and passenger flow. The aim is to analyse the algorithm performance.

The rest of the paper is organized as follows. In Sect. 2 is given an ant colony optimization algorithm. In Sect. 3 a brief discussion on InterCriteria Analysis background is done. In Sect. 4 the transportation problem is formulated and an ACO algorithm which solves it is proposed. Experimental results are shown and analysed in Sect. 5. In Sect. 6 are drawn some concluding remarks and possibilities for future work.

2 Ant Colony Optimization Method

The considered optimization problem (see Section III) is NP-hard, and therefore we consider the use of a meta-heuristic search for its solution. Therefore is impractical to be applied some traditional numerical method. Hereof we apply Ant Colony Optimization (ACO) algorithm, one of the best metaheuristics.

The behaviour of ants in nature has inspired the creation of this method. Ants put on the ground chemical substance called pheromone, which help them to return to their nest when they look for a food. The ants smell the pheromone and follow the path with a highest pheromone concentration. Thus they find shorter path between the nest and the source of the food.

The ACO algorithm uses a colony of artificial ants that behave as cooperating agents, like ants in the nature. With the help of the pheromone they try to construct better solutions and to optimize them. The problem is represented by a graph and the solution is represented by a path in the graph or by tree in the graph. The graph representation is crucial for the good algorithm performance.

Ants start from random nodes of the graph and try to construct feasible solutions. When all ants construct their solution the pheromone values are updated. Ants compute a set of feasible moves and select the best one, according to the transition probability rule. The transition probability p_{ij}, to choose the node j when the current node is i, is based on the heuristic information η_{ij} and on the pheromone level τ_{ij} of the move, where $i, j = 1, \ldots, n$. α and β shows the importance of the pheromone and the heuristic information respectively.

$$p_{ij} = \frac{\tau_{ij}^{\alpha}\,\eta_{ij}^{\beta}}{\sum\limits_{k \in \{allowed\}} \tau_{ik}^{\alpha}\,\eta_{ik}^{\beta}} \tag{1}$$

The construction of the heuristic information function depends highly of the solved problem. It is appropriate combination of problem parameters and is very important for ants' management. An ant selects the move with highest probability. The initial pheromone is set to a small positive value τ_0 and then ants update this value after completing the construction stage [13, 16, 18]. The search stops when $p_{ij} = 0$ for all values of i and j, which means that it is impossible to include new node in the current partial solution.

The pheromone trail update rule is given by:

$$\tau_{ij} \leftarrow \rho\tau_{ij} + \Delta\tau_{ij}, \tag{2}$$

where $\Delta\tau_{ij}$ is a new added pheromone and it depends of the quality of achieved solution.

The pheromone is decreased with a parameter $\rho \in [0, 1]$. This parameter models evaporation in the nature and decreases the influence of old information in the search process. After that, a new pheromone is included. It is proportional to the quality of the solution (value of the fitness function). Several variants of ACO algorithm exist. The main difference is the pheromone updating.

Multi-Objective Optimization (MOP) begins in the nineteenth century in the work of Edgeworth and Pareto in economics [23]. The optimal solution for MOP is not a single solution as for mono-objective optimization problems, but a set of solutions defined as Pareto optimal solutions. A solution is Pareto optimal if it is not possible to improve a given objective without deteriorating at least another one. The main goal of the resolution of a multi-objective problem is to obtain the Pareto optimal set and consequently the Pareto front. One solution dominates another if minimum one of its components is better than the same component of other solutions and other components are not worse. The Pareto front is the set of non dominated solutions related to the solved problem. After that, the users decide which solution from the Pareto front to use according additional constraints, related with their specific application. When metaheuristics are applied, the goal becomes to obtain solutions close to the Pareto front.

3 InterCriteria Analysis

According to [2, 3, 5, 7], we will obtain an Intuitionistic Fuzzy Pair (IFP) as the degrees of "agreement" and "disagreement" between two criteria applied on different objects. An IFP is an ordered pair of real non-negative numbers $\langle a, b\rangle$ such that:

$$a + b \leq 1.$$

Let us be given an IM [6] whose index sets for rows consist of the names of the criteria and for columns—objects. We will obtain an IM with index sets consisting of the names of the criteria both for rows and for columns. The elements IFPs of this

IM corresponds to the degrees of "agreement" and degrees of "disagreement" of the considered criteria.

The following two points are supposed:

1. All criteria provide an evaluation for all objects and all these evaluations are available.
2. All the evaluations of a given criteria can be compared amongst themselves.

Further, by O we denote the set of all objects $O_1, O_2, \ldots, O_n$ being evaluated, and by $C(O)$ the set of values assigned by a given criteria C to the objects, i.e.

$$O \stackrel{\text{def}}{=} \{O_1, O_2, \ldots, O_n\},$$

$$C(O) \stackrel{\text{def}}{=} \{C(O_1), C(O_2), \ldots, C(O_n)\}.$$

Let $x_i = C(O_i)$. Then the following set can be defined:

$$C^*(O) \stackrel{\text{def}}{=} \{\langle x_i, x_j\rangle | i \neq j \;\&\; \langle x_i, x_j\rangle \in C(O) \times C(O)\}.$$

In order to find the degrees of "agreement" of two criteria the vector of all internal comparisons of each criteria is constructed. This vector fulfil exactly one of the following three relations—R, $\overline{R}$ and $\tilde{R}$. For a fixed criterion C and any ordered pair $\langle x, y\rangle \in C^*(O)$ it is required:

$$\langle x, y\rangle \in R \Leftrightarrow \langle y, x\rangle \in \overline{R} \tag{3}$$

$$\langle x, y\rangle \in \tilde{R} \Leftrightarrow \langle x, y\rangle \notin (R \cup \overline{R}) \tag{4}$$

$$R \cup \overline{R} \cup \tilde{R} = C^*(O) \tag{5}$$

From the above it is seen that We only need to consider a subset of $C(O) \times C(O)$ for the effective calculation of $V(C)$ (vector of internal comparisons). From Eqs. (3)–(5) it follows that if we know what is the relation between x and y, we also know what is the relation between y and x. Thus, we will only consider lexicographically ordered pairs $\langle x, y\rangle$.

Let:

$$C_{i,j} = \langle C(O_i), C(O_j)\rangle.$$

We construct the vector with exactly $\frac{n(n-1)}{2}$ elements:

$$V(C) = \{C_{1,2}, C_{1,3}, \ldots, C_{1,n}, C_{2,3}, C_{2,4}, \ldots, C_{2,n},$$

$$C_{3,4}, \ldots, C_{3,n}, \ldots, C_{n-1,n}\}.$$

for a fixed criterion C.

Further, we replace the vector $V(C)$ with $\hat{V}(C)$, where for each $1 \leq k \leq \frac{n(n-1)}{2}$ for the k-th component it is true:

$$\hat{V}_k(C) = \begin{cases} 1 \text{ iff } V_k(C) \in R, \\ -1 \text{ iff } V_k(C) \in \overline{R}, \\ 0 \text{ otherwise.} \end{cases}$$

We determine the degree of "agreement" ($\mu_{C,C'}$) between the two criteria as the number of matching components. This can be done in several ways, e.g. by counting the matches or by taking the complement of the Hamming distance. The degree of "disagreement" ($\nu_{C,C'}$) is the number of components of opposing signs in the two vectors. This may also be done in various ways.

It is obvious that:

$$\mu_{C,C'} = \mu_{C',C},$$

$$\nu_{C,C'} = \nu_{C',C},$$

and $\langle \mu_{C,C'}, \nu_{C,C'} \rangle$ is an IFP.

The difference

$$\pi_{C,C'} = 1 - \mu_{C,C'} - \nu_{C,C'} \tag{6}$$

is considered as a degree of "uncertainty".

4 Problem Formulation

Various problems arise in the area of long-distance passenger transport with a different kind of transport. One of the problem is optimal scheduling [20], others concern the optimal management of the passenger flow [24]. In some developments, it is involved only one type of vehicle [14]. The common is that all they are difficult in computational point of view.

Our problem concerns passengers travelling in a same direction, covered with several different types of vehicles, trains and buses and every one of them can have different price and speed. The problem is how passengers will be allocated to different vehicles Let the first stop be station A and the last stop be station B. There are two kinds of vehicles, trains and buses, which travel between station A and station B. Every vehicle has its set of stations where it stops, only the first station and the terminus are common for all vehicles. Some of the stations can be common for some of the vehicles. Let the set of all stations is $S = \{s_1, \ldots, s_n\}$ and on every station s_i, $i = 1, \ldots, n-1$, n is the number of stations, at every time slot there are number of passengers which want to travel to station s_j, $j = i+1, \ldots, n$. Every vehicle travel with different speed and the price to travel from station s_i to station s_j can be different. We fix a parameters k_1 and k_2. They are used for calculation of the time

and price intervals respectively. If a passenger have in mind to start his travel at time t he will chose a vehicle in the interval $(t - k_1, t + k_1)$. If a passenger have in mind to pay for his travel price P he can pay price from the interval $(P, P + P * k_2/100)$. Thus, we include in our model some fuzzy element with an aim it to become more realistic.

The input data of our problem are set of stations S, starting time of every vehicle from the first station, time for every vehicle to go from station s_i to station s_j, the capacity of every vehicle, the price for every vehicle to travel from one station to another one, number of passengers which want to travel from one station to another one at every moment. Our algorithm calculates how many passengers will get on every of the vehicles on station s_i to station s_j at every time slot. There are two objectives, the total price of all tickets, Eq. 3, and the total travel time, Eq. 4. If some vehicle does not stop on some station, we put the travel time and the price to this destination to be 0.

$$TP = \sum_{i=1}^{M} p_i \tag{7}$$

where TP is the total price, M is the number of passengers, p_i is the price, paid by the passenger i.

$$TT = \sum_{i=1}^{M} T_i \tag{8}$$

where TT is the total time, M is the number of passengers, T_i is the traveling time of passenger i.

The output is the number of passengers in every vehicle in every station and the values of the two objective functions.

It is NP-hard multi-objective optimization problem, therefore we chose a meta-heuristic method to solve it, in particular ACO.

The model is prepared to solve the problem for one direction. It can be applied to model and optimize transportation network direction by direction. One of the important points of the ACO algorithm is representation of the problem by graph. In our case the time is divided to time periods, $N \times 24$ time periods correspond to 60/N min, thus $2 \times 24 = 48$ time periods, correspond to 30 min. Every station is represented by set of $N \times 24$ nodes, showing different time moments in which a vehicle stops on this station. The pheromone is deposited on the nodes of the graph. The ants start to construct their solutions from the first station. If the number of the passengers from this station is P, the ants chose a random number P_1 from the interval $[0, min\{P, C_1\}]$ and assign this number to the first vehicle as a number of passengers. To the next vehicle the interval is decreased with P_1. C_1 is the capacity of the vehicle. The number of all passengers getting vehicle in some time moment is maximal possible. If there is only one vehicle at this moment the maximal possible number of passengers gets on this vehicle. We model the number of the passengers

for the next stations by applying probabilistic rule called transition probability. Our heuristic information is a sum of the reciprocal values of the two objective functions.

5 Results and Discussion

5.1 Experimental Solutions

We have programmed our ACO algorithm in C programming language. After several experiments the algorithm parameters are set as it is shown in a Table 1.

We test our algorithm on one real problem, destination Sofia Varna. The starting station is Sofia, Bulgarian capital and the terminus is Varna the maritime capital of the country. The distance between the first and the last station is about 450 km. There are 5 trains and 23 buses which travel from Sofia to Varna, but they move with different speed, the prices are different and they stop on different stations between Sofia and Varna. There are not data available on passenger numbers therefore we approximate them, taking in to account the population of every one of the towns where some of the vehicles stops. 5 trains and 23 buses, with different speed and price travel between them every day. The stations can differ for different vehicles.

The Tables 2 and 3 shows achieved solutions by two variant of ACO algorithm, deterministic and fuzzy respectively. The results in Table 2 are from our previous work [19] where we apply the deterministic variant of the algorithm. 10 ants are used and the algorithm is run 100 iterations. In a both cases there are 5 nondominated

Table 1 ACO parameters

ρ	0.5
α	1
β	1
τ_0	0.5
Number of ants	10
Number of iterations	100

Table 2 Experimental results Sofia Varna, deterministic

No	Price	Time	Train
1	51843	25840	1951
2	51797	25842	1952
3	51579	25862	1978
4	51571	25869	1979
5	51563	25870	1980

Table 3 Experimental results Sofia Varna, fuzzy

No	Price	Time	Train
1	51821	25856	1961
2	51775	25864	1963
3	51565	25873	1991
4	51560	25880	1995
5	51549	25882	1998

solutions. In every row are shown the travel price of hall passengers, the travel time of hall passengers and the sum of the passengers used train. In the both tables can be seen that the solutions with more passengers in the train have more travelling time and less price. The number of passengers used train or respectively bus is changed if on the same station on the same time there is more than one transportation possibility for deterministic case. In deterministic case the difference in number of passengers in the train comes from long destinations. In fuzzy variant of the algorithm we observe that the number of the passengers in the train is more than in the bus comparing with deterministic case. When the price between the bus and train is similar in a short destination in the fuzzy case it is perceived as the same, it is the same for the time, and the passengers chose bus or train with the same probability. In deterministic case even the small difference is perceived as a different and the vehicle with less price has high probability to be chosen by the passengers which prefer cheaper transportation. Thus we can explain why in the fuzzy case more passengers chose the train than in deterministic one.

5.2 InterCriteria Analysis of the Results

The cross-platform software for ICrA approach, ICrAData, is used [17]. The input index matrices for ICrA have the form of Tables 4 and 8.

In Table 4 the number of passengers used train for long destination (more than 300 km) are presented. In Table 8 the results from number of passengers used train for any destination are presented.

In index matrices presented in Tables 5 and 6 the results of ICrA based on solutions listed in Table 4 are presented. The obtained ICrA results are analysed based on the proposed in [4] consonance and dissonance scale. The scheme for defining the consonance and dissonance between each pair of criteria is presented in Table 7.

As can be seen the results show that the different solutions are in positive or in strong positive consonance, i.e., there is a high correlation between them (see Table 5). Moreover, there is no significant degree of "uncertainty" in the data. The proposed ACO algorithm performs very well.

Table 4 Number of passengers used train for long destination

No	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
1	30	30	30	30	30
2	60	60	60	60	60
3	210	210	210	210	210
4	30	25	7	30	30
5	60	60	59	60	60
6	210	210	209	210	210
7	30	30	30	30	30
8	60	60	60	60	60
9	208	187	206	209	210
10	30	30	30	30	30
11	60	60	60	60	60
12	210	210	210	210	210
13	30	30	30	30	30
14	60	60	60	60	60
15	210	210	210	210	210

Table 5 Index matrix for $\mu_{C,C'}$ (Intuitionistic fuzzy estimations over data in Table 4)

$\nu_{C,C'}$	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
Sol 1	1	0.9619	0.8952	1	0.9619
Sol 2	0.9619	1	0.9333	0.9619	0.9238
Sol 3	0.8952	0.9333	1	0.8952	0.8571
Sol 4	1	0.9619	0.8952	1	0.9619
Sol 5	0.9619	0.9238	0.8571	0.9619	1

Table 6 Index matrix for $\nu_{C,C'}$ (Intuitionistic fuzzy estimations over data in Table 4)

$\nu_{C,C'}$	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
Sol 1	0	0	0	0	0
Sol 2	0	0	0	0	0
Sol 3	0	0	0	0	0
Sol 4	0	0	0	0	0
Sol 5	0	0	0	0	0

Table 7 Consonance and dissonance scale [4]

Interval of $\mu_{C,C'}$	Meaning
[0.00–0.05]	Strong negative consonance
(0.05–0.15]	Negative consonance
(0.15–0.25]	Weak negative consonance
(0.25–0.33]	Weak dissonance
(0.33–0.43]	Dissonance
(0.43–0.57]	Strong dissonance
(0.57–0.67]	Dissonance
(0.67–0.75]	Weak dissonance
(0.75–0.85]	Weak positive consonance
(0.85–0.95]	Positive consonance
(0.95–1.00]	Strong positive consonance

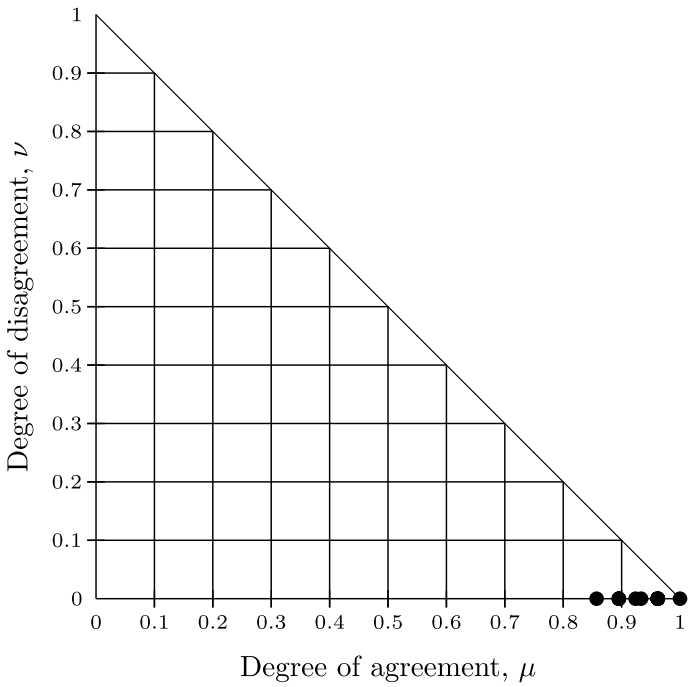


Fig. 1 Presentation of ICrA results in the intuitionistic fuzzy interpretation triangle

The obtained ICrA results are visualized on Fig. 1 within the specific triangular geometrical interpretation of IFSs, thus allowing us to order these results according simultaneously to the degrees of "agreement" $\mu_{C,C'}$ and "disagreement" $\nu_{C,C'}$ of the intuitionistic fuzzy pairs.

Table 8 Number of passengers used train for any destination

No	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
1	10	10	10	10	10
2	20	20	20	20	20
3	70	70	70	70	70
4	30	30	30	30	30
5	60	60	60	60	60
6	210	210	210	210	210
7	7	25	30	30	30
8	59	60	60	60	60
9	209	210	210	210	210
10	10	10	10	10	10
11	20	20	20	20	20
12	70	70	70	70	70
13	30	30	30	30	30
14	60	60	60	60	60
15	206	187	208	209	210
16	30	30	30	30	30
17	60	60	60	60	60
18	210	210	210	210	210
19	10	10	10	10	10
20	20	20	20	20	20
21	70	70	70	70	70
22	10	10	10	10	10
23	20	20	20	20	20
24	70	70	70	70	70
25	10	10	10	10	10
26	70	70	70	70	70
27	30	30	30	30	30
28	60	60	60	60	60
29	210	210	210	210	210

Table 9 Index matrix for $\mu_{C,C'}$ (Intuitionistic fuzzy estimations over data in Table 8)

$\nu_{C,C'}$	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
Sol 1	1	0.9606	0.9507	0.9507	0.9409
Sol 2	0.9606	1	0.9901	0.9901	0.9803
Sol 3	0.9507	0.9901	1	1	0.9901
Sol 4	0.9507	0.9901	1	1	0.9901
Sol 5	0.9409	0.9803	0.9901	0.9901	1

Table 10 Index matrix for $\nu_{C,C'}$ (Intuitionistic fuzzy estimations over data in Table 8)

$\nu_{C,C'}$	Sol 1	Sol 2	Sol 3	Sol 4	Sol 5
Sol 1	0	0.0222	0.0222	0.0222	0.0222
Sol 2	0.0222	0	0	0	0
Sol 3	0.0222	0	0	0	0
Sol 4	0.0222	0	0	0	0
Sol 5	0.0222	0	0	0	0

The results from ICrA over data in Table 8 (Tables 9 and 10) are identical. The different solutions are in positive or in strong positive consonance, i.e., there is a high correlation between them. Again, there is no significant degree of "uncertainty" in the data. This result proves that the algorithm is suitable for this transportation modeling problem.

6 Conclusion

Transportation is a very important branch of economics and our everyday life. The different kinds of transportation propose different services. Ones are faster, others are cheaper. The passenger decision depends on his preferences. In this paper we propose a model of the flow of passengers taking into account the two main criteria that guide the passengers in their choice, travelling time and travelling price. Thus the problem is defined as multi-objective optimization problem with two objective functions. A fuzzy variant of the model is proposed. When the prices or times are in a predefined interval, they are considered equal. Thus the model becomes closer to human thinking and, from there, more realistic.

The applied InterCriteria Analysis over numerical results obtained from ACO algorithm show that the proposed algorithm performs very well and it is a proof that the algorithm is suitable for this problem.

The proposed model can help for transport analysis of existing transport. It can predict the change of passenger flow when some vehicle is included or excluded and when the timetable is changed. Thus the transportation can be optimized and to become close to the people's needs. In a future we can include additional elements in the model like other preferences of the passengers.

Acknowledgements Work presented here is partially supported by the National Scientific Fund of Bulgaria under grant DFNI DN12/5 "Efficient Stochastic Methods and Algorithms for Large-Scale Problems", Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program.

References

1. El Amaraoui, A., Mesghouni, A.K.: Train scheduling networks under time duration uncertainty. In: Proceedings of the 19th World Congress of the International Federation of Automatic Control, pp. 8762–8767 (2014)
2. Atanassov, K., Mavrov, D., Atanassova, V.: Intercriteria decision making: a new approach for multicriteria decision making, based on index matrices and intuitionistic fuzzy sets. Issues IFSs GNs **11**, 1–8 (2014)
3. Atanassov, K.: On Intuitionistic Fuzzy Sets Theory. Springer, Berlin (2012)
4. Atanassov, K., Atanassova, V., Gluhchev, G.: InterCriteria analysis: ideas and problems. Notes on Intuitionistic Fuzzy Sets **21**(1), 81–88 (2015)
5. Atanassov, K.: Intuitionistic fuzzy sets, VII ITKR session, Sofia, 20–23 June 1983, reprinted. Int. J. Bioautomation **20**(S1), S1–S6 (2016)
6. Atanassov, K.: On index matrices, Part 1: standard cases. Adv. Stud. Contemp. Math. **20**(2), 291–302 (2010)
7. Atanassov, K.: Review and new results on intuitionistic fuzzy sets, mathematical foundations of artificial intelligence seminar, Sofia, 1988, Preprint IM-MFAIS-1-88, Reprinted. Int. J. Bioautom. **20**(S1), S7–S16 (2016)
8. Atanassov, K., Vassilev, P.: On the intuitionistic fuzzy sets of n-th type. In: Gaweda A., Kacprzyk J., Rutkowski L., Yen G. (eds.), Advances in data analysis with computational intelligence methods. Studies in Computational Intelligence, vol. 738, pp. 265–274. Springer, Cham (2008)
9. Atanassova, V., Mavrov, D., Doukovska, L., Atanassov, K.: Discussion on the Threshold Values in the InterCriteria Decision Making Approach, Notes on Intuitionistic Fuzzy Sets, vol. 20, No. 2, pp. 94–99 (2014)
10. Atanassova, V., Doukovska, L., Atanassov, K., Mavrov, D.: Intercriteria decision making approach to EU member states competitiveness analysis. In: Shishkov, B. (ed.), Proceedings of the International Symposium on Business Modeling and Software Design—BMSD'14, pp. 289–294 (2014)
11. Angelova, M., Roeva, O., Pencheva, T.: InterCriteria analysis of crossover and mutation rates relations in simple genetic algorithm. Proceedings of the Federated Conference on Computer Science and Information Systems **5**, 419–424 (2015)
12. Assad, A.A.: Models for rail transportation. Transp. Res. Part A Gen. **14**(3), 205–220 (1980)
13. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press (1999)
14. Diaz-Parra, O., Ruiz-Vanoye, J.A., Loranca, B.B., Fuentes-Penna, A., Barrera-Camara, R.A.: A survey of transportation problems. J. Appl. Math. **2014**, Article ID 848129, 17 pages (2014)
15. Dong, C.H., Xiong, Z.H., Shao, C.H., Zhang, H.: A spatial-temporal-based state space approach for freeway network traffic flow modelling and prediction. J. Transportmetrica A: Trans. Sci. **11**(7), 574–560 (2015)
16. Dorigo, M., Stutzle, T.: Ant Colony Optimization. MIT Press (2004)
17. Ikonomov, N., Vassilev, P., Roeva, O.: ICrAData—software for intercriteria analysis. Int. J. Bioautom. **22**(1), 1–10 (2018)
18. Fidanova, S., Atanasov, K.: Generalized net model for the process of hibride ant colony optimization. Comptes Randus de l'Academie Bulgare des Sci. **62**(3), 315–322 (2009)
19. Fidanova, S.: Metaheuristic Method for Transport Modelling and Optimization, Studies in Computational Intelligence, vol. 648, pp. 295–302. Springer (2016)
20. Hanseler, F.S., Molyneaux, N., Bierlaire, M., Stathopoulos, A.: Schedule-based estimation of pedestrian demand within a railway station. In: Proceedings of the Swiss Transportation Research Conference (STRC), 14–16 May 2014
21. Jin, J.G., Zhao, J., Lee, D.H.: A column generation based approach for the train network design optimization problem. J. Trans. Res. **50**(1), 1–17 (2013)
22. Marinov, E., Vassilev, P., Atanassov, K.: On separability of intuitionistic fuzzy sets. In: Novel Developments in Uncertainty Representation and Processing, Advances in Intelligent Systems and Computing, vol. 401. Springer, Cham, pp. 111–123 (2106)

23. Mathur, V.K.: How well do we know pareto optimality? J. Econ. Educ. **22**(2), 172–178 (1991)
24. N. Molyneaux, F. Hanseler, M. Bierlaire, Modelling of train-induced pedestrian flows in railway stations. In: Proceedings of the Swiss Transportation Research Conference (STRC), 14–16 May 2014
25. Roeva, O., Fidanova, S., Paprzycki, M.: InterCriteria analysis of ACO and GA hybrid algorithms. Stud. Comput. Intell. **610**, 107–126 (2016)
26. Roeva, O., Fidanova, S., Vassilev, P., Gepner, P.: InterCriteria analysis of a model parameters identification using genetic algorithm. Proc. Fed. Conf. Comput. Sci. Inf. Syst. **5**, 501–506 (2015)
27. Todinova, S., Mavrov, D., Krumova, S., Marinov, P., Atanassova, V., Atanassov, K., Taneva, S.G.: Blood plasma thermograms dataset analysis by means of intercriteria and correlation analyses for the case of colorectal cancer. Int. J. Bioautomation 20(1), 115–124 (2016)
28. Traneva, V., Atanassova V., Tranev, S.: Index matrices as a decision-making tool for job appointment. In: Nikolov, G. et al. (eds.), Springer Nature Switzerland AG, NMA 2018, vol. 11189, 1–9, 2019. LNCS (2018)
29. Vassilev, P., Todorova, L., Andonov, V.: An auxiliary technique for intercriteria analysis via a three dimensional index matrix. Notes on Intuitionistic Fuzzy Sets, vol. 21, No. 2, pp. 71–76 (2015)
30. Vassilev, P., Ribagin, S.: A note on intuitionistic fuzzy modal-like operators generated by power mean. In: Kacprzyk, J., Szmidt, E., Zadrożny, S., Atanassov, K., Krawczak, M. (eds.), Advances in Fuzzy Logic and Technology 2017. EUSFLAT 2017, IWIFSGN 2017. Advances in Intelligent Systems and Computing, vol. 643, pp. 470–475. Springer, Cham (2018)
31. Vassilev, P.: A Note on New Distances between Intuitionistic Fuzzy Sets, Notes on Intuitionistic Fuzzy Sets, vol. 21, No. 5, pp. 11–15 (2015)
32. Woroniuk, C., Marinov, M.: Simulation modelling to analyze the current level of utilization of sections along rail rout. J. Trans. Lit. **7**(2), 235–252 (2013)

Approximation and Exact Algorithms for Multiprocessor Scheduling Problem with Release and Delivery Times

Natalia Grigoreva

Abstract The multiprocessor scheduling problem is defined as follows: set of jobs have to be executed on parallel identical processors. For each job we know release time, processing time and delivery time. At most one job can be performed on every processor at a time, but all jobs may be simultaneously delivered. Preemption on processors is not allowed. The goal is to minimize the time, by which all tasks are delivered. Scheduling tasks among parallel processors is a NP-hard problem in the strong sense. The best known approximation algorithm is Jackson's algorithm, which generates the list schedule by selecting the ready job with the largest delivery time. This algorithm generates no delay schedules. We define an IIT (inserted idle time) schedule as a feasible schedule in which a processor can be idle at a time when it could begin performing a ready job. The paper proposes the approximation inserted idle time algorithm for the multiprocessor scheduling. We proved that deviation of this algorithm from the optimum is smaller then twice the largest processing time. Then by combining the MDT/IIT algorithm and the branch and bound method this paper presents BB algorithm which can find optimal solutions for the problem. To illustrate the efficiency of our approach we compared two algorithms on randomly generated sets of jobs.

1 Introduction

We consider the problem of scheduling jobs with release and delivery times on parallel identical processors.

We consider a set of jobs $U = \{u_1, u_2, \ldots, u_n\}$. For each job we know its processing time $t(u_i)$, its release time $r(u_i)$ the time at which the job is ready for performing and its delivery time $q(u_i)$. All data are integer. Set of jobs is performed on m parallel

N. Grigoreva (✉)
St. Petersburg State University, Universitetskaja nab. 7/9, 199034 St. Petersburg, Russia

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_7

identical processors. Any processor can run any job and it can perform no more than one job at a time. Preemption is not allowed. The schedule defines the start time $\tau(u_i)$ of each job $u_i \in U$. The makespan of the schedule S is the quantity

$$C_{\max} = \max\{\tau(u_i) + t(u_i) + q(u_i) | u_i \in U\}.$$

The goal is to minimize $C_{\max}$, the time by which all jobs are delivered. Following the classification scheme proposed by Graham et al. [1], this problem is denoted by $P|r_i, q_i|C_{max}$.

The problem is equivalent to model $P|r_i|L_{\max}$ with due dates $d(u_i)$, rather than delivery times $q(u_i)$. The equivalence is shown by replacing each delivery time $q(u_i)$ by due date $d(u_i) = q_{\max} - q(u_i)$, where $q_{\max} = \max\{q(u_i) \mid u_i \in U\}$. In this problem the objective is to minimize the maximum lateness of jobs $L_{\max} = \max\{\tau(u_i) + t(u_i) - d(u_i) | u_i \in U\}$.

This problem relates to the scheduling problem [2], very similar problems can arise in different application fields [3]. The problem plays the main role in some important applications, for example, in the Resource Constrained Project Scheduling Problem [2], and it is NP-hard [4].

The single machine problem with release and delivery times is denoted by $1|r_j, q_j|C_{max}$ and it is NP-hard too [4]. The $1|r_j, q_j|C_{\max}$ is also a main component of several more complex scheduling problems, such that flowshop and jobshop scheduling [5, 6] and uses in real industrial application [6]. The problem $1|r_j, q_j|C_{\max}$ has been studied by many researches [7–10].

The problem $P|r_j, q_j|C_{\max}$ is a generalization of the single-machine scheduling problem with release and delivery times $1|r_j, q_j|C_{\max}$. The problem arises as a strong relaxation of the multiprocessor flow shop problem [11]. The problem has been the subject of numerous papers, some of these works focus on problems with a precedence constrains [12].

Most of these studies have focused to obtain lower bounds [13, 14], the development of exact solution of the problem [6, 15] or a polynomial time approximation scheme (PTAS) [8, 16].

However, despite its practical importance, only Jackson's algorithm is used as a simple list heuristic algorithm for the $P|r_j, q_j|C_{\max}$.

The worst-case performance of Jackson's algorithm has been investigated by Gusfield [17] and Carlier [15]. Gusfield [17] examined Jackson's heuristic for the problem to minimize the maximum lateness of jobs with release times and due dates and proved that difference between the lateness given by Jackson's algorithm and the optimal lateness is bounded by $(2m - 1)t_{max}/m$ and this bound is tight.

Carlier [15] proved that $C_{\max} - C_{opt} \leq 2t_{\max} - 2$, where $C_{\max}$ is the objective function of Jackson's rule schedule, and C_{opt} is the optimal makespan.

Gharbi and Haouari [18] proposed improved Jackson's algorithm which uses an $O(n \log n)$-time preprocessing procedure in order to reduce the number of jobs to be scheduled and investigated its worst-case performance.

The preprocessing procedure can be briefly described as follows. Let $j(k)$ is the job with the kth smallest release time. A condition which allows to define the start

time of a job $j_0 \in \{j_1, j_2, ..., j_m\}$ at $r(j_0)$ in an optimal schedule is $r(j_0) + t(j_0) = \min\{r(j_k) + t(j_k) | k \in 1..m\} \leq r(j_{m+1})$. Then a job j_0 can be deleted from the set of jobs. This deleting rule is recursively applied to the new jobset $U \backslash \{j_0\}$. Let U_r be the set of jobs deleted according to this rule. Then the above deleting rule can be applied to the reversing problem (where by reversing the roles of the release and delivery times). Let U_q be the set of jobs deleted according to this second rule.

Therefore, the problem can be solved on a reduced jobset, denoted by UJ. Let S_{UJ} is a feasible schedule with makespan equal to $C_{max}(S_{UJ})$. Then the improved Jackson's algorithm constructs a complete schedule with makespan equal to $C_{\max} = \max\{C_{\max}(S_{UJ}), \max(r_j + t_j + q_j | j \in U_r \cup U_q)\}$.

Most of research in scheduling is devoted to the development of nondelay schedule. A nondelay schedule has been defined by Baker [19] as a feasible schedule in which a processor cannot be idle at a time when it could start performing a ready job. Kanet and Sridharam [20] defined an inserted idle time schedule (IIT)as a feasible schedule in which a processor can idle, if there is the ready job and reviewed the literature with problem setting where IIT scheduling may be required. Most of papers considered problem with single processor. It is known that an optimal schedule can be IIT schedule. Therefore,it is important to develop algorithms that can build IIT schedule.

In [21] we considered multiprocessor scheduling problem with precedence constrained and proposed the branch and bound algorithm, which use an inserted idle time algorithm for m parallel identical processors.

In [22] we investigated the inserted idle time algorithm for single machine scheduling with release times and due dates.

The goal of this paper is to propose an approximation and exact IIT algorithms for $P|r_j, q_j|C_{max}$ problem. First we propose an approximation algorithm and investigate its worst-case performance.Then by combining the MDT/IIT algorithm and the branch and bound method this paper presents $B\&B$ algorithm which can find optimal solution for the problem.

In order to confirm the effectiveness of our approach we tested our algorithms on randomly generated examples.

First in Sect. 2, we propose an approximation IIT algorithm named MDT/IIT (maximum delivery time/inserted idle time). In Sect. 3 we investigate the worst-case performance of MDT/IIT algorithm. In Sect. 4 we present the branch and bound method. In Sect. 5 we present the results of testing the algorithm. Summary of this paper is in Sect. 6.

2 Approximation Algorithm MDT/IIT

Algorithm MDT/IIT generates the schedule, in which a processor can be idle at the time when it could begin performing a job.

Let $r_{\min} = \min\{r(u_i) \mid u_i \in U\}$ and $q_{\min} = \min\{q(u_i) \mid u_i \in U\}$.

First we calculate the lower bound LB of the optimal makespan [15]:
$LB = \max\{r_{\min} + \sum_{i=1}^{n} t(u_i)/m + q_{\min}, \max\{r(u_i) + t(u_i) + q(u_i) \mid u_i \in U\}\}$.

Let $t_{\max} = \max\{t(u_i) \mid u_i \in U\}$.

Let a partial schedule S_k have been constructed, where k is the number of scheduling jobs. We know the start time $\tau(u_i)$ of each job u_i from S_k and the processor that performs it. Let $C_{\max}(S_k)$ be the makespan of S_k.

Let $time_k[l]$ be the time of completion of all jobs of partial solution S_k assigned to processor l.

Procedure $SET(l, u_j, k, C_{\max}(S_k))$ sets a job u_j on processor l at step k and includes the job u_j in S_k.

$SET(l, u_j, k, C_{\max}(S_k))$.

1. $\tau(u_j) := \max\{time_k[l], r(u_j)\}$.
2. $k := k + 1$.
3. $time_k[l] := \tau(u_j) + t(u_j)$.
4. $C_{\max}(S_k) := \max\{C_{\max}(S_{k-1}), \tau(u_j) + t(u_j) + q(u_j)$.

The approximation schedule S is constructed by MDT/IIT algorithm as follows:

1. Determine the processor l_0 such that
$$t_{\min}(l_0) = \min\{time_k[i] | i \in 1..m\}.$$
2. If there is no job u_i, such that $r(u_i) \le t_{\min}(l_0)$ then
$t_{\min}(l_0) := \min\{r(u_i) \mid u_i \notin S_k\}$.
3. Select a job u with the largest delivery time
$q(u) = \max\{q(u_i) \mid r(u_i) \le t_{\min}(l_0)\}$.
4. If $t_{\min}(l_0) > t_{\max}$ then $SET(l_0, u, k, C_{\max}(S_k))$; go to 11.
5. Select a job u^* such that
$q(u^*) = \max\{q(u_i) \mid t_{\min}(l_0) < r(u_i) < t_{\min}(l_0) + t(u)\}$.
6. If there is no such job u^* or one of inequality is hold
$q(u) \ge q(u^*)$ or $q(u^*) \le LB/3$, or $r(u^*) \ge t_{\max}$
then $SET(l_0, u, k, C_{\max}(S_k))$. Go to 11.
$idproc(l_0) = r(u^*) - t_{min}(l_0)$.
7. If $q(u^*) - q(u) < idproc(l_0)$, then $SET(l_0, u, k, C_{\max}(S_k))$. Go to 11.
8. Select a job u_1 which can be executed during the time interval $[t_{\min}(l_0), r(u^*)]$, namely such that
$q(u_1) = \max\{q(u_i) \mid t_{\min}(l_0) \ge r(u_i) \;\&\; t(u_i) \le idle(u^*)\}$.
If job u_1 exists, then $SET(l_0, u1, k, C_{\max}(S_k))$. Go to 11.
9. Select the ready job u_2 such that
$q(u_2) = \max\{q(u_i) \mid t_{\min}(l_0) < r(u_i) \;\&\; r(u_i) + t(u_i) \le r(u^*)\}$.
If we find u_2, then $SET(l_0, u2, k, C_{\max}(S_k))$. Go to 11.
10. $SET(l_0, u^*, k, C_{\max}(S_k))$.
11. If $k < n$, then go to 1.
12. If $k = n$, we construct the approximation schedule $S = S_n$ and we have the objective function $C_{\max}(S) = C_{\max}(S_n)$.

The algorithm sets on the processor l_0 the job u^* with the largest delivery time $q(u^*)$. If job u^* is not ready, then the processor l_0 does not work in the interval $[t_1, t_2]$, where $t_1 = t_{\min}(l_0)$, $t_2 = r(u^*)$.

In order to avoid too much idle of the processor the inequality $q(u^*) - q(u) \geq idproc(l_0)$ is verified on step 7 and if it is hold, we select job u^*. In order to use the idle time of the processor l_0 we look for job u_1 or u_2 to perform in this interval (see steps 8 and 9). Job u^* starts at $\tau(u^*) = r(u^*)$.

The MDT/IIT algorithm generates the schedule in $O(mn^2)$ times. It generates the schedule by n iterations, the processor selection requires $O(m)$ times and the job selection requires $O(n)$ time on each iteration.

3 Property of MDT/IIT Algorithm

Let algorithm generate a schedule S, and for each job j we have the start time $\tau(j)$. The makespan is $C_{\max}(S) = \max\{\tau(j) + t(j) + q(j) \mid j \in U\}$.

Definition 1 Critical job j_c is the first processed job such that $C_{\max}(S) = \tau(j_c) + t(j_c) + q(j_c)$.

Let C_{opt} be the length of an optimal schedule.

Theorem 1 $C_{\max}(S) - C_{opt} < t_{\max}(2m - 1)/m$, *and this bound is tight.*

Proof Let c be the critical job then $C_{\max}(S) = \tau(c) + t(c) + q(c)$. If the processors do not idle in the time interval $[0, \tau(c)]$, then we set $\tau^* = 0$, else

$$\tau^* = \max\{t \mid 0 < t < \tau(c)\},$$

where t is the time, when the number of processors working from time $t - 1$ to t is smaller then m.

Let $J = \{v_i \in U | \tau^* \leq \tau(v_i) < \tau(c)\}$ be the set of jobs, which begin in interval $[\tau^*, \tau(c))$.

Let $\tau(j_0) = \max\{\tau(v_i) | \tau(j_0) < \tau(c) \ \& \ q(v_i) < q(c)\}$. The job j_0 is the last scheduling job with $q(j_0) < q(c)$ and $\tau(j_0) < \tau(c)$. If there is no such work j_0, then we set $\tau(j_0) = 0$.

We consider four cases.

Case 1. There is not any idle time of processors before $\tau(c)$ and then $\tau^* = 0$.

Let $\tau(j_0) = 0$, then all jobs, which start time $\tau(v_i) < \tau(c)$, have delivery time $q(v_i) \geq q(c)$. The jobs from the set J must start in interval $[0, \tau_c)$, then

$$\sum_{v_i \in J} t(v_i) \geq m\tau(c)$$

and

$$C_{opt} \geq \sum_{v_i \in J} t(v_i)/m + t(c)/m + q(c) \geq \tau(c) + t(c)/m + q(c).$$

Then

$$C_{\max}(S) - C_{opt} \leq t(c) - t(c)/m < t_{\max}.$$

Case 2. Let $0 \leq \tau(j_0) < \tau^* < t_{\max}$.
Then $q(v_i) \geq q(c), \forall v_i \in J$.

We can consider three sets of jobs:

$A_1 = \{v_i \in J | r(v_i) \geq \tau^*\}$, the jobs can start in interval $[\tau^*, \tau_c)$,

$A_2 = \{v_i \in J | r(v_i) < \tau^*\}$, the jobs can start before τ^*,

$$A_3 = \{v_i \in U | \tau(v_i) \leq \tau^* - 1 \ \& \ \tau(v_i) + t(v_i) \geq \tau^*\}.$$

A_3 contains not more $m - 1$ jobs and this jobs process in the time interval $[\tau^* - 1, \tau^*]$.

There are no any idle time of processors in the interval $[\tau^*, \tau(c)]$, then

$$T_A = \sum_{v_i \in A_3} (t(v_i) - 1) + \sum_{v_i \in A_1} t(v_i) + \sum_{v_i \in A_2} t(v_i) \geq m(\tau(c) - \tau^*).$$

The jobs from set A_1 can process only after the time τ^*, but the jobs from sets A_2 and A_3 can process before τ^*. The job c can process before τ^*, if $r(c) < \tau^*$.

$$C_{opt} \geq (T_A + t(c))/m + q(c) \geq \tau(c) - \tau^* + t(c)/m + q(c).$$

Hence

$$C_{\max}(S) - C_{opt} \leq \tau^* + t(c) - t(c)/m < t_{\max}(2 - 1/m),$$

because $\tau^* < t_{max}$ (see step 3 of MDT/IIT algorithm).

Case 3. Let $t_{\max} \leq \tau^*$ and $\tau(j_0) < \tau^*$.

If $t_{\max} \leq \tau^*$ then $A_2 = \emptyset$ and the job c can process only after τ^*. Then

$$\sum_{v_i \in A_3} (t(v_i) - 1) + \sum_{v_i \in A_1} t(v_i) \geq m(\tau(c) - \tau^*).$$

$$C_{opt} \geq \tau^* + \sum_{v_i \in A_1} t(v_i)/m + t(c)/m + q(c) \geq$$

$$\geq \tau(c) - \sum_{v_i \in A_3} (t(v_i) - 1)/m + t(c)/m + q(c).$$

A_3 contains not more $m - 1$ jobs, hence

$$C_{\max}(S) - C_{opt} \leq t(c) - t(c)/m + 1/m \sum_{v_i \in A_3} (t(v_i) - 1) \leq$$

$$\leq t(c) - t(c)/m + (m-1)/m(t_{\max} - 1) \leq (2t_{\max} - 1)(m-1)/m < t_{\max}(2 - 1/m).$$

Case 4. Consider the case $0 \leq \tau^* \leq \tau(j_0)$.

Let $J = \{v_i \in U | \tau(j_0) < \tau(v_i) < \tau(c)\}$.

For all $v_i \in J$ it is true, that $r(v_i) > \tau(j_0)$, otherwise the processor must process job v_i instead of j_0 and $q(v_i) \geq q(c)$.

Then

$C_{opt} \geq \tau(j_0) + 1 + \sum_{v_i \in J} t(v_i)/m + t(c)/m + q(c)$.

We can see the set of jobs:

$A_3 = \{v_i \in U | \tau(v_i) \leq \tau(j_0) \,\&\, \tau(v_i) + t(v_i) \geq \tau(j_0) + 1\}$, the jobs must process in interval $[\tau(j_0), \tau(j_0) + 1]$. The set A_3 contains m jobs. Then

$$\sum_{v_i \in A_3} (t(v_i) - 1) + \sum_{v_i \in J} t(v_i) \geq m(\tau(c) - \tau(j_0) - 1).$$

$$C_{opt} \geq \tau(j_0) + 1 + \tau(c) - \tau(j_0) - 1 - 1/m \sum_{v_i \in A_3} (t(v_i) - 1) +$$

$$+ t(c)/m + q(c) = \tau(c) + t(c)/m + q(c) - 1/m \sum_{v_i \in A_3} (t(v_i) - 1).$$

A_3 contains m jobs, hence

$$C_{\max}(S) - C_{opt} \leq 1/m \sum_{v_i \in A_3} (t(v_i) - 1) + t(c)(m-1)/m \leq$$

$$\leq t_{max} - 1 + t_{\max}(m-1)/m.$$

Hence

$$C_{\max}(S) - C_{opt} \leq t_{\max}(2m-1)/m - 1.$$

Now, we show that this bound is tight and for this purpose consider the example.

Example 1 Consider the $m^2 + m + 1$ jobs and m processors instance. There are $2m$ jobs $v_i : r(v_i) = 0$; $t(v_i) = m$; $q(v_i) = 0$. There are $m(m-1)$ jobs $u_i : r(u_i) = m - 1$; $t(u_i) = 1$; $q(u_i) = m$ and the job $a : r(a) = m - 1$; $t(a) = m$; $q(a) = m$.

The makespan of MDT/IIT schedule is $C_{\max}(MDT) = 5m - 2$. The makespan of the Jackson's schedule is $C_{\max}(JR) = 4m - 1$. The optimal makespan is equal $3m$.

Table 1 shows the schedule posted by algorithm MDT/IIT, and Table 2 shows the optimal schedule, for the case $m = 3$. The first row of the table shows the time of

Table 1 MDT schedule $C_{\max}(MDT) = 5m - 2$

t	$m-1$	$m-1$		m	m	m
$P1$	idle	u_1	u_4	a	v_3	$v6$
$P2$	idle	u_2	u_5	v_1	v_4	idle
$P3$	idle	u_3	u_6	v_2	v_5	idle

Table 2 Optimal schedule $C_{\max} = 3m$

t	m	m			m
$P1$	v_3	a			v_6
$P2$	v_1	u_1	u_4	u_3	v_4
$P3$	v_2	u_2	u_5	u_6	v_5

the assignments. The next three lines indicate the tasks performed on the processors $P1$, $P2$, $P3$, respectively. □

We can see that $C_{\max}(MDT) - C_{opt}$ is equal $2m - 2$, that is $2t_{max} - 2$.

We compare schedules constructed by MDT/IIT algorithm with schedules constructed by nondelay Jackson's algorithm. Consider next example.

Example 2 Consider the $m^2 + 1$ jobs and m processors instance.

There are m jobs $v_i : r(v_i) = 0; t(v_i) = m; q(v_i) = 0$. There are $m(m-1)$ jobs $u_i : r(u_i) = 1; t(u_i) = 1; q(u_i) = m$ and there is job $a : r(a) = 1; t(a) = m; q(a) = m$.

The makespan of the Jackson's schedule is $C_{\max}(JR) = 4m - 1$. The makespan of MDT/IIT schedule is $C_{\max}(MDT) = 3m$. The makespan of an optimal schedule is $C_{opt} = 2m + 1$.

Table 3 shows the schedule posted by algorithms MDT/IIT, Table 4 shows the Jackson's schedule schedule and Table 5 shows the optimal schedule for the case $m = 3$.

The algorithms JR and MDT are in a certain sense opposites: if the algorithm JR generates a schedule with a large error, the algorithm MDT/IIT works well and vice versa. Examples 1 and 2 illustrate this property of the algorithms. We propose the combined algorithm CA that builds two schedules: one by the algorithm JR, the other by the algorithm MDT and selects the best.

Table 3 MDT schedule $C_{\max}(MDT) = 3m$

t	1	$m-1$		m	m
$P1$	idle	u_1	u_4	a	v_3
$P2$	idle	u_2	u_5	v_1	idle
$P3$	idle	u_3	u_6	v_2	idle

Table 4 The Jackson's schedule $C_{\max}(JR) = 4m - 1$

t	m	$m-1$		m
$P1$	v_1	u_1	u_4	a
$P2$	v_2	u_2	u_5	idle
$P3$	v_3	u_3	u_6	idle

Table 5 Optimal schedule $C_{\max} = 2m + 1$

t	1	m			m
$P1$	idle	a			v_6
$P2$	idle	u_1	u_4	u_3	v_4
$P3$	idle	u_2	u_5	u_6	v_5

4 Branch and Bound Method for $P|r_i, q_i|C_{\max}$

First we must give a definition of partial solutions σ_k, where k is the number of jobs, in the Branch and bound method. We know the start time $\tau(u_i)$ of each job $u_i \in \sigma_k$ and the processor that performs it. We know $time_k[l]$ the time of completion of all jobs of σ_k assigned to processor l, for $l \in 1 : m$ and $I(\sigma_k)$ - the total idle time of processors in partial schedule σ_k.

For a partial solution σ_k we know $t_{\min}(k) = \min\{time_k[l] \mid l \in 1 : m\}$, and a set of jobs $U(\sigma_k)$, which we call the ready jobs. These are tasks that need to add to a partial solution σ_k, so check all the possible continuation of the partial solutions.

The pseudo-code of Branch and bound method $BB(U, m; S, C_{\max}(S))$ is shown in Algorithm 1.

The algorithm takes a set of jobs U and the number of processors m and produces the best solution S and makespan $C_{\max}(S)$. We describe the steps of the algorithm in detail.

First the approximation solution S_{best} is generated by CA algorithm and upper bound is calculated $UB := C_{\max}(S_{best})$. The upper bound UB is equal the makespan of the best known schedule S_{best} and is renewed whenever a new schedule S_{new} with $C_{\max}(S_{new}) < UB$ is obtained (step 1).

The lower bound LB is computed by Lower bound procedure (step 2). If $LB = UB$ the optimal solution have been constructed and the algorithm stops (steps 3, 4).

In the steps 6–8 the root vertex of the search tree is initialized. The empty schedule σ_0 is added into the set of active partial solutions APS, and the total idle time of processors $I(UB)$ is calculated.

The loop (steps 9–23 is repeated until $APS <> \emptyset$.

The algorithm selects the partial solution $\sigma_k \in APS$ that has the smallest lower bound $LB(\sigma_k)$ (step 10).

Then It generates child partial solutions σ_{k+1} of the partial solution σ_k, according branching rule IIT (step 11).

If algorithm generates a full schedule S, the makespan $C_{\max}(S)$ is calculated. If $C_{\max}(S) < UB$ then the upper bound UB is updated and all partial solutions, which lower bound is greater or equal new UB, are eliminated (steps 12–18).

A lower bound of all new partial solutions $LB(\sigma_k + 1)$ is calculated (step 20) according Lower bound procedure. All partial solutions, which lower bound $LB(\sigma_k + 1)$ is greater or equal UB, are eliminated (step 21).

4.1 Branching Rule IIT

For each vertex in search tree (each partial solution σ_k) there is a set of ready jobs.

Definition 2 Job $u \notin \sigma_k$ is called the ready job for σ_k, if $r(u)$ satisfies the inequality $r(u) \leq t_{min}(\sigma_k) - I(UB) + Idle(\sigma_k)$.

Let the set $U(\sigma_k)$ be the set of ready jobs.

Branching rule IIT selects all ready jobs u for partial solution σ_k, to generate partial solutions $\sigma_{k+1} = \sigma_k \cup u$.

1. Determine the processor l_0 such that

$$t_{\min}(l_0) = \min\{time_k[i] | i \in 1..m\}.$$

2. Select a job u with the largest delivery time $q(u) = \max\{q(u_i) \mid u_i \in U(\sigma_k)\}$.
3. $\tau(u) := \max\{t_{\min}(l_0), r(u)\}$.
4. $time_k[l] := \tau(u) + t(u)$.
5. Calculate the idle time of the processor l_0 before the start of job u^* $idproc(l_0) = r(u) - t_{\min}(l_0)$.
6. If $idproc(l_0) > 0$ then calculate the total idle time of all processor $Idle(\sigma_{k+1}) := Idle(\sigma_k) + idproc(l_0)$.

4.2 The Idle Time of All Processor $I(UB)$

Let $I(UB)$ be the total value of idle of processors if upper bound is equal UB. Then

$$I(UB) = (UB - 1 - q_{\min})m - \sum_{i=1}^{n} t[u_i].$$

4.3 Lower Bound Procedure

We calculate the lower bound LB for each new partial solution σ_k.

Let $q_{\min}(\sigma_k) = \min\{q(u_i) \mid u_i \notin \sigma_k\}$. Let $C_{\max}(\sigma_k) = \max\{\tau(u) + t(u) + q(u) \mid u_i \in \sigma_k\}$.

For all jobs $u \notin \sigma_k$ we calculate new release time $r_{new}(u) = \max\{t_{\min}(\sigma_k), r(u)\}$. The first lower bound is equal $LB_1 = \max\{r_{new}(u) + t(u_i) + q(u_i) \mid u_i \notin \sigma_k\}$

The second lower bound:

$$LB_2 = r_{\min} + \left[\left(\sum_{i=1}^{n} t(u_i) + Idle(\sigma_k)\right)/m\right] + q_{\min}(\sigma_k).$$

Then $LB^* = \max\{C_{\max}(\sigma_k), LB_1, LB_2\}$.

The third lower bound is calculated similarly to the method proposed in [23].

Let be y the number different new release times for jobs $u \notin \sigma_k$ then $r^{(1)} < r^{(2)} < \cdots < r^{(y)}$. Put $r^{(y+1)} := LB^*$.

We consider the set Ω of time intervals $[r^{(i)}, r^{(j)}]$, where $t_{\min}(k) \le r^{(1)}$, and $1 \le i < j \le y+1$.

For all jobs $u \notin \sigma_k$ we calculate $d(u)$ the latest start time then $d(u) = LB^* - q(u) - t(u)$.

Algorithm calculates $\alpha(u)[r^{(i)}, r^{(j)}]$ the minimal time for job u in the time interval $[r^{(i)}, r^{(j)}]$.

If job u can be performed outside this interval, then $\alpha(u)[r^{(i)}, r^{(j)}]$ is equal to zero.

Then if $(d(u) - t(u) \le r^{(i)})$ or $(r^{(j)} \le r_{new}(u))$ then $\alpha(u)[r^{(i)}, r^{(j)}] := 0$

else

$$\alpha(u)[r^{(i)}, r^{(j)}] := \min\{t(u), r^{(j)} - r^{(i)}, d(u) + t(u) - r^{(i)}, r^{(j)} - r_{new}(u)\}.$$

Let $MJ([r^{(i)}, r^{(j)}])$ be the total time of all jobs in the time interval $[r^{(i)}, r^{(j)}]$, then

$$MJ([r^{(i)}, r^{(j)}]) = \sum_{u \notin \sigma_k} \alpha(u)[r^{(i)}, r^{(j)}].$$

Let $MP([r^{(i)}, r^{(j)}])$ be the total time of processors in the time interval $[r^{(i)}, r^{(j)}]$, then

$$MP([r^{(i)}, r^{(j)}]) = \sum_{l=1}^{m} \max\{0, r^{(j)} - \max\{r^{(i)}, time_k(l)\}\}.$$

Let's find the busiest interval in which the difference between the total running time $MJ([r^{(i)}, r^{(j)}])$ and the available processor power $MP([r^{(i)}, r^{(j)}])$ is maximum.

$est = \max\{MJ([r^{(i)}, r^{(j)}]) - MP([r^{(i)}, r^{(j)}]) \mid [r^{(i)}, r^{(j)}] \in \Omega\}$.

If $est > 0$ then $LB^* := LB^* + \lceil est/m \rceil$.

Then $LB(\sigma_k) = LB^*$.

Algorithm 1 BB/IIT algorithm

```
Construct a schedule S by algorithm CA; UB := C_max(S), S_best := S;
Compute LB
if LB = UB then
   GOTO 24
end if
Initialize root vertex of search tree: Set time[i] = 0; i ∈ 1 : m; σ_0 = ∅;
Compute I(UB) = (UB − 1 − q_min)m − Σ_{i=1}^{n} t[u_i].
APS = σ_0
while APS <> ∅ do
   Select partial solution σ_k ∈ APS
   Generate a set of child partial solutions to σ_k
   if k+1=n then
      Calculate C_max(σ_n)
      if C_max(σ_n) < UB then
         UB := C_max(σ_n); S_best := σ_n
         Eliminate all partial solutions from APS such that LB(σ_i) ≥ UB
         Calculate I(UB)
      end if
   else
      Compute LB(σ_{k+1}) for all child partial solutions
      Eliminate child partial solutions if LB(σ_{k+1}) ≥ UB
   end if
end while
We find optimal schedule S_best and its makespan is equal C_max(S_best).
```

4.4 Elimination Rule

We eliminate all partial solutions $\sigma_k \in APS$ such that $LB(\sigma_k) \geq UB$.

5 Computation Result

In this section we present the results of testing the proposed algorithm on several types of tests. The quality of the schedules we estimated the average relative gap produced by each algorithm, where the gap is equal to $RT = (C_{\max} - LB)/LB$. We compared algorithms JR, MDT/IIT and the combined algorithm CA, that builds two schedules (one schedule by the algorithm JR, the other by the algorithm MDT) and selects the best solution. We tested the branch and bound algorithm BB/MDT too. The program for the MDT and BB/MDT algorithms is coded in Object Pascal and

Table 6 Type A. Variation of n

n	$RT(MDT)$	$RT(JR)$	$N_{opt}(CA)$	$RT(CA)$
100	0.219	0.228	0	0.216
200	0.147	0.159	0	0.141
300	0.061	0.066	0	0.058
400	0.053	0.051	0	0.052
500	0.047	0.042	0	0.039

compiled with Delphi 7. All experiments were performed out on a laptop computer 1.9 GHz speed and 4 GB memory.

At first, we studied MDT/IIT algorithm.

The experiment considered several types of examples. The number of jobs n changed from 100 to 500.

In examples type A job processing time, release and delivery times are generated with discrete uniform distributions between 1 and n. Groups for $m = 20$ and $n = 100, 200, 300, 400, 500$ were tested. For each n we generate 30 instances. 150 instances of type A are tested. The results are given in Table 6. The first column of this table contains the number of jobs n.The columns $N_{opt}(MDT)$, $N_{opt}(JR)$ and $N_{opt}(CA)$ shows the cases (in percents) where optimal schedules were obtained by MDT method, JR method and combined method.

We can see that the problem becomes easier as n increases, because the average number of jobs per processor tends to increase. The average relative gap ranges from 4 to 21% for CA algorithm. The combined algorithm allows to improve RT in all cases.

In the next experiment we fix the number of jobs $n = 500$ and change the number of processors m from 3 to 170. For each m we generate 30 instances and a total of 240 instances are tested. The results of the experiments are shown in Table 7. The first column of this table contains the number of processors m. Table 7 shows the performance of JR, MDT and CA algorithms.

Table 7 Type A. Variation of m

m	$N_{opt}(MDT)$	$RT(MDT)$	$N_{opt}(JR)$	RT(JR)	$N_{opt}(CA)$	$RT(CA)$
3	0	0.003	0	0.004	0	0.002
10	0	0.019	0	0.016	0	0.014
20	0	0.042	0	0.046	0	0.041
30	0	0.068	0	0.075	0	0.066
50	0	0.146	0	0.135	0	0.132
100	0	0.201	0	0.195	0	0.191
130	0	0.021	0	0.025	0	0.019
170	98	0.001	97	0.001	**100**	**0.000**

Table 7 shows that average relative gap increases when m changes from 3 to 100 and reaches a maximum at $m = 100$. Then it decreases and when $m = 170$ algorithm MDT generates 98% optimal solutions, algorithm JR 96% and algorithm CA generates optimal solutions for all instances. Algorithms JR and MDT give very close solutions and only with $m = 3, 20, 30, 130, 170$ the algorithm MDT has an advantage. The combined algorithm allows to improve RT in all cases.

We can see from Tables 6 and 7 that the most difficult examples occur when the average number of jobs per processor is equal 5.

In the next series of tests, we restricted our instances to those types that found hard. The number of jobs n is equal to 100 and the number of processors m is equal to 20 (5 jobs on average per processor). In instances of type C we change $t_{\max}$. Type C, that were randomly generated as follows: the job processing time is generated with discrete uniform distributions between 1 and $t_{\max}$, where t_{max} changes from 20 to 500. For each $t_{\max}$ we generate 30 instances.240 instances of type C are tested. Release and delivery times are generated with discrete uniform distributions on [1,100]. The results of the work are given in Table 8.

We can see that the problem becomes more difficult with increasing $t_{\max}$, the average relative gap increases and remains large at a $t_{\max}$ from 100 to 500. The maximum deviation is reached at $t_{\max} = 200$. The combined algorithm allows to increase (at $t_{\max} = 50$) the number of optimal solutions by 9% and to improve RT in all cases.

In the series of tests considered, the average deviation was slightly different for the algorithms JR and MDT. The combined algorithm allowed us to slightly improve the value of the objective function.

In order to get a better picture of the actual effectiveness of MDT/IIT we consider other types of instances.

In next series we consider instances in which jobs have the same processing time.

Type EJ (Equal job): The heads are drawn from the discrete uniform distribution on [1, 10] and tails from [1, 60], $n = 100$. All processing times $t_i = 60$. We can see the computational results in Table 9, where the last column F contains the difference $RT(JR) - RT(MDT)$.

Table 8 Type C. Variation of $t_{\max}$

$t_{\max}$	$N_{opt}(MDT)$	$RT(MDT)$	$N_{opt}(JR)$	RT(JR)	$N_{opt}(CA)$	$RT(CA)$
20	100	0.000	99	0.000	100	0.000
50	51	0.004	52	0.005	61	0.003
70	0	0.016	0	0.014	0	0.013
100	0	0.212	0	0.219	0	0.210
200	0	0.223	0	0.221	0	0.220
300	0	0.209	0	0.218	0	0.207
400	0	0.207	0	0.213	0	0.206
500	0	0.203	0	0.206	0	0.202

Table 9 Type EJ. Variation of m

m	$RT(MDT)$	$RT(JR)$	$RT(CA)$	F
20	0.03	0.04	0.03	0.01
30	0. 22	0.24	0.22	0.02
40	**0.26**	0.32	0.26	0.06
50	**0.17**	0.40	**0.17**	0.23

We can see that for examples of Type EJ the average relative gap is less for algorithm MDT/IIT for all values of m. For $m = 50$, the average relative gap for the JR algorithm is equal 0.40, but for the MDT/IIT algorithm it is only 0.17.

Type SG (small-great) : The heads are generated from the discrete uniform distribution on [1, 10] and tails on [1, 80], $n = 100$. The processing times are drawn from the discrete uniform distribution on [40, 60].

Table 10 shows the results of examples of type SG. For cases $m = 40$ and $m = 50$, there is a significant difference between the results obtained by different algorithms. The average relative gap for MDT algorithm and JR algorithm is equal 0.14 and 0.24, respectively, for $m = 40$. Algorithm MDT/IIT generated 100% of optimal solutions, whereas algorithm JR only 25% for $m = 50$. We observe from Tables 9 and 10 that MDT/IIT exhibits a good performance with instances of types EJ and SL.

Type GS (great-small): The $r(u)$ are drawn from the discrete uniform distribution on [1, 100] and $q(u)$ on [1, 20], $n = 100$. Table 11 shows the results of examples of type GS. The processing times are drawn from the discrete uniform distribution on $[1, n]$.

Table 10 Type SG. Variation of m

m	$RT(MDT)$	RT(JR)	$RT(CA)$	F
20	0.10	0.11	0.10	0.01
30	0.19	0.23	0.19	0.04
40	**0.14**	0.24	**0.14**	0.10
50	**0.000**	0.23	**0.000**	0.23

Table 11 Type GS. Variation of m

m	$RT(MDT)$	$RT(JR)$	$RT(CA)$
3	0.015	0.014	0.014
10	0.100	0.110	0.092
20	0. 239	0.236	0.227
30	0.205	0.198	0.192
50	**0.006**	0.008	**0.000**

Table 12 Relative error of makespan for BB/MDT method for type A

n	m	N_{opt}	$RT < 0.05$	$RT < 0.1$	$RT > 0.1$
100	10	66.1	13.2	11.9	7.8
100	20	72.7	9.8	7.6	9.9
200	10	79.2	7.9	5.4	7.5
200	20	76.3	11.5	8.1	4.1
300	10	78.1	9.5	8.4	4.0
300	20	82.9	7.3	2.7	7.1
Average		75.88	9.86	7.35	6.73

For examples of the type GS, the greatest deviation is observed at $m = 20$ and $m = 30$. The optimal solutions were obtained only at $m = 50$. The combined algorithm works better than each of the algorithms separately in all types of examples.

We investigated the branch and bound algorithm BB/MDT. The program running time was limited to 60 s. The average relative error $RT = (C_{max} - LB)/LB$ of schedules obtained by BB/MDT algorithm are presented in next tables.

In examples type A job processing time, release and delivery times are generated with discrete uniform distributions between 1 and n. Groups for $m = 10, 20$ and $n = 100, 200, 300$ were tested. For each n and m we generate 30 instances. 180 instances of type A are tested. Table 12 shows the results for BB/MDT algorithm. The column N_{opt} shows the cases (in percents) where optimal schedules were obtained by this method. The next column shows the number of cases (in percents) in which approximate solutions within the error of 0.05 were obtained, but optimal solutions could not be obtained because of the time limit. But an intermediate solution can be an optimal solution. The next two columns shows the number of cases in which $RT \in (0.05, 0.1]$ and RT greater then 0.1.

It is seen from Table 12 that optimal solutions were obtained for 75,88% (in average) of the cases tested. For 85.74% of the cases approximate solutions having error of less than 5% were obtained.

In the next series of tests, we consider instances to type C. As we saw from experiments with an approximate algorithm MDT/IIT for fixed n and m, the largest deviation from the lower bound is observed with increasing $t_{\max}$. The number of jobs n is equal to 100 and the number of processors m is equal to 20 (5 jobs on average per processor). In instances of type C we change $t_{\max}$ from 70 to 500. For each $t_{\max}$ we generate 30 instances. 180 instances of type C are tested.

The results of the work are given in Table 13.

The branch-and-bound method gets fewer optimal solutions (with the same CPU time limit) as the maximum job processing time increases. As $t_{\max}$ increases from 70 to 500, the number of optimal solutions decreases from 71.1 to 61.6%, and the number of solutions for which $RT < 0.05$ decreases from 79.9 to 74.7%.

Table 13 Relative error of makespan for BB/MDT method for type C

$t_{\max}$	$N_{opt}(MDT)$	$RT < 0.05$	$RT < 0.1$	$RT > 0.1$
70	71.1	8.8	10.5	9.7
100	68.3	9.7	13.3	8.7
200	69.4	8.6	11.9	10.1
300	63.8	11.6	10.7	13.9
400	64.5	10.4	9.9	13.2
500	61.6	13.1	11.1	9.2

6 Conclusion

We propose an approximation IIT algorithm named MDT/IIT (maximum delivery time/ inserted idle time) for $P|r_j, q_j|C_{max}$ problem.

We proved that $C_{\max}(S) - C_{opt} < t_{\max}(2m-1)/m$, and this bound is tight, where $C_{\max}$ is the objective function of MDT/IIT schedule, and C_{opt} is the makespan of an optimal schedule.

We observe that MDT/IIT algorithm exhibits a good performance with instances in which delivery times are large compared with processing times and release times.

We propose the combined algorithm that builds two schedules (one by the algorithm JR, the other by the algorithm MDT) and selects the best solution.The algorithms JR and MDT are in a certain sense opposites: if the algorithm JR generates a schedule with a large error, the algorithm MDT works well and vice versa.

Computational experiments have shown that the combined algorithm works better than each of the algorithms separately. We investigated the brand and bound algorithm BB/MDT. Optimal solutions were obtained for 75,88% (in average) of the cases tested. For 85.74% of the cases approximate solutions having error of less than 5% were obtained.

References

1. Graham, R.L., Lawner, E.L., Rinnoy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling. A surv. Ann. Disc. Math. **5**(10), 287–326 (1979)
2. Brucker, P.: Scheduling Algorithms, 5th edn. Springer, Berlin (2007)
3. J. Omer, A. Mucherino, Referenced vertex ordering problem. Theory, Applications and Solution Methods HAL Open Archives, hal-02509522, version 1 (Mar 2020)
4. Ullman, J.: NP-complete scheduling problems. J. Comput. Syst. Sci. **171**, 394–394 (1975)
5. Artigues, C., Feillet, D.: A branch and bound method for the job-shop problem with sequence-dependent setup times. Ann. Oper. Res. **159**, 135–159 (2008)
6. Chandra, C., Liu, Z., He, J., Ruohonen, T.: A binary branch and bound algorithm to minimize maximum scheduling cost. Omega **42**, 9–15 (2014)
7. Carlier, J.: The one machine sequencing problem. Eur. J. Oper. Res. **11**, 42–47 (1982)

8. Hall, L.A., Shmoys, D.B.: Approximation schemes for constrained scheduling problems. In: Proceedings of the 30th IEEE Symposium on Foundations of Computer Science, pp. 134 —139 (1989)
9. Nowicki, E., Smutnicki, C.: An approximation algorithm for a single-machine scheduling problem with release times and delivery times. Discret. Appl. Math. **48**, 69–79 (1994)
10. Potts, C.N.: Analysis of a heuristic for one machine sequencing with release dates and delivery times. Oper. Res. **28**(6), 445–462 (1980)
11. Carlier, J., Nèron, E.: An exact algorithm for solving the multiprocessor flowshop. RAIRO Oper. Res. **34**, 1–25 (2000)
12. Zinder, Y.: An iterative algorithm for scheduling UET tasks with due dates and release times. t Eur. J. Oper. Res. **149**, 404–416 (2003)
13. Carlier, J., Pinson, E.: Jackson's pseudo preemptive schedule for the $Pm|rj, qj|Cmax$ scheduling problem. Ann. Oper. Res. **83**, 41–58 (1998)
14. Haouari, M., Gharbi, A.: Lower bounds for scheduling on identical parallel machines with heads and tails. Ann. Oper. Res. **129**, 187–204 (2004)
15. Carlier, J.: Scheduling jobs with release dates and tails on identical machines to minimize the makespan. Eur. J. Oper. Res. **29**, 298–306 (1987)
16. Mastrolilli, M.: Efficient approximation schemes for scheduling problems with release dates and delivery times. J. Sched. **6**, 521–531 (2003)
17. Gusfield, D.: Bounds for Naive multiple machine scheduling with release times and deadlines. J. Algorithms **5**, 1–6 (1984)
18. Gharbi, A., Haouari, M.: An approximate decomposition algorithm for scheduling on parallel machines with heads and tails. Comput. Oper. Res. **34**, 868 —883 (2007)
19. Baker, K.R.: Introduction to Sequencing and Scheduling. Wiley, New York (1974)
20. Kanet, J., Sridharan, V.: Scheduling with inserted idle time: problem taxonomy and literature review. Oper. Res. **48**(1), 99–110 (2000)
21. Grigoreva, N.S.: Branch and bound method for scheduling precedence constrained tasks on parallel identical processors. In: Proceedings of the World Congress on Engineering 2014, WCE 2014 London, UK. Lecture Notes in Engineering and Computer Science, pp. 832–836 (2014)
22. Grigoreva, N.,: Single machine inserted idle time scheduling with release times and due dates. In: Proceedings of DOOR2016, Vladivostoc, Russia. 19–23 Sept 2016. CEUR-WS, vol. 1623, pp. 336—343 (2016)
23. Fernandez, E., Bussell, B.: Bounds the number of processors and time for multiprocessor optimal schedules. IEEE Trans. Comput. **4**(11), 745–751 (1973)

A Hybrid Method for Scheduling Multiprocessor Tasks on Two Dedicated Processors

Méziane Aïder, Fatma Zohra Baatout, and Mhand Hifi

Abstract In this paper, we investigate the use of a hybrid method for approximately solving the problem of scheduling multiprocessor tasks on two dedicated processors. Such a problem is NP-hard and its goal is to provide the best order of available tasks assigned either to one processor, or simultaneously to two processors such that the execution of the last assigned task should be minimized. Starting with a solution provided by a knapsack procedure, the algorithm combines a series of operators oscillating between both intensification and diversification strategies. For each internal/current solution, a reactive search is used for enhancing the quality of that solution. In order to highlight the global search space, a drop/rebuild strategy is incorporated around the best solution achieved. Finally, the performance of the hybrid method is experimentally analyzed on a set of benchmark instances, where its achieved results are compared to those reached by the best method available in the literature. New results have been provided.

1 Introduction

The scheduling problem is a very important problem in factory scheduling (cf. Hoogeveen et al. [9]). Most of existing researches consider the processing time of each operation, the job's transportation to another machine for the next operation, the setup needed to process the next job, etc. In addition, the times associated to these steps may increase the complexity of the problem to solve. In this paper, we tackle Scheduling multiprocessor Tasks on Two dedicated Processors (noted ST2P); that is

M. Aïder · F. Z. Baatout
LaROMaD, USTHB, BP 32 El Alia, 16111 Alger, Algérie
e-mail: m-aider@usthb.dz

F. Z. Baatout
e-mail: fbaatout@usthb.dz

M. Hifi (✉)
EPROAD EA 4669, Université de Picardie Jules Verne, 7, rue du Moulin Neuf, Amiens, France
e-mail: hifi@u-picardie.fr

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_8

an NP-hard combinatorial optimization problem. The goal of ST2P is to assign all available tasks to either a single processor or two different processors. The studied problem can be viewed as a special case of the scheduling family, where the set of tasks is divided into three subsets:

(i) A first subset of available tasks should be performed on the first processor,
(ii) A second subset of tasks represents those assigned on the second processor and,
(iii) A third subset containing tasks which must be performed simultaneously on both processors.

One can observe that, on the one hand, several scheduling problems may consider different objective functions:

(i) minimizing the makespan that corresponds to minimize the completion time of the last executed task,
(ii) to minimize the summation of the delays of all tasks,
(iii) to minimize both delays and makespan, etc.

On the other hand, several versions of the scheduling problem can be accessed (i) on the number of available processors, (ii) on how tasks are assigned on certain processors, etc.

In this paper, ST2P is tackled where its goal is to minimize the completion time of the last executed task (namely makespan). Such a problem can be encountered in several real-world applications, like production and data transfer (cf. Manna and Chu [11]). An instance of ST2P is defined as follows: let N denote the set containing n tasks to scheduling on two dedicated processors (namely P_1 and P_2) such that a task j is released at time r_j and has to be processed without preemption during its processing time p_j and, C_j is the completion time of the task j while $C_{\max}$ denotes the makespan of the schedule to minimize. As described in Graham et al. [7], ST2P is defined as $P2|fix_j, r_j|C_{\max}$, such that

- $P2$: denotes two processors on which all tasks must be executed.
- fix_j: means that task j is assigned to both processors.
- r_j: is the release date of task j.
- p_j: denotes the processing time related to task j when executed on the processors.
- $C_{\max}$: is the makespan (completion time) of the last assigned/executed task.

The remainder of the paper is organized as follows. Section 2 reviews some works related to the scheduling problem. Section 3 describes the proposed hybrid method for approximately solving ST2P. ST2P's tight lower bound, proposed by Manna and Chu [11], is described in Sect. 3.1, where it has been used for analyzing the quality of the results provided by all methods. A starting solution, using a knapsack greedy rule, is described in Sect. 3.2. The intensification operators, combined with a tabu list, are described in Sect. 3.3. The diversification strategy, using the drop and rebuild operator, is discussed in Sect. 3.4. Section 4 exposes the experimental analysis of the proposed method on a set of benchmark instances, where its provided results are compared to those achieved by the best method available in the literature.

2 Background

As discussed in Brucker [5], the scheduling problems family contains many problem types. The performance measures of these problems are often categorized into three main groups of criteria: (i) those based on completion time, (ii) those based on due dates and, (iii) those based on inventory cost and utilization. Due to the NP-hardness of the problem tackled in this paper, there are few available papers studying it in the literature.

Bianco et al. [2] tackled the problem of scheduling tasks on two dedicated processors with preemptive constraints (noted $P2|fix_j, r_j, pmtn|C_{\max}$), where the task can be interrupted and can be completed later. The authors designed an optimal solution procedure that is based on a two steps polynomial time complexity.

Blazewicz et al. [4] tackled a special case of scheduling multiprocessor tasks on two identical parallel processors. The authors discussed the complexity analysis for special cases, like considering (i) the scheduling with unit execution time, (ii) the preemptable tasks with ready times and, due-dates and, (iii) precedence constraints. For these cases, the goal of the problem is to minimize the schedule length and the maximum lateness.

Thesen [12] designed a tabu search-based algorithm for approximately solving multiprocessor scheduling problems. The proposed algorithm combines tabu strategy, local search operator, and how to manage the lists used. Several strategies have been considered, like random blocking related to the size of the tabu list, frequency-based penalties for diversifying the search process, and the hashing operator for stocking high solutions. The experimental part showed that some combinations have better behavior than others; in this case, the achieved results, on benchmark instances, are better than those reached by several algorithms of the literature.

Blazewicz et al. [3] tackled the problem of scheduling multiprocessor tasks on three dedicated processors. The authors made a complexity analysis of the problem and studied different cases related to this problem for which they proposed optimal solutions in polynomial time complexity.

Buffet et al. [6], developed two tabu search-based algorithms for solving the multiprocessor scheduling problem using m processors. The authors followed the standard principle of the tabu search, where a starting solution is built by respecting a legal schedule, the intensification strategy that checks possible permutations between tasks for improving the quality of the solutions, the diversification strategy using a local search for exploring unvisited subspaces. The resulting algorithm was tested on thirty randomly generated instances and showed that the method could outperform one of the best methods of the literature.

Concerning the problem studied in this paper, scheduling tasks on two dedicated processors, Manaa and Chu [11] proposed an exact algorithm to solve it. The algorithm is based upon the classical branch-and-bound procedure, where the internal nodes are bounded with special lower and upper bounds. The experimental part showed the performance of such a method, where it was able to solve instances up to thirty tasks within fifty minutes.

Kacem and Dammak [10] tailored an effective genetic algorithm for approximately solving the scheduling of a set of tasks on two dedicated processors. The principle of the algorithm is based upon the classical genetic principle reinforced with a constructive procedure able to provide feasible solutions for the problem. The resulting algorithm was evaluated on random instances generated following Manaa and Chu's [11] generator, where the experimental part showed that the method was able to achieve solution values closest to those provided by the tight lower bound proposed by Manaa and Chu [11].

Because the proposed method can be viewed as an extended version of the method proposed in Aïder et al. [1] and, in order to make the paper self-containing, some parts of the aforementioned paper are repeated in what follows.

3 Tackling the ST2P with a Hybrid Method

In this section, we first present a tight lower bound used for evaluating the performance of the proposed method and that of the best method available in the literature. Second and last, we present the principle of the method and describe all procedures employed by the method.

3.1 ST2P's Lower Bound

Manaa and Chu [11] proposed a nice lower bound for ST2P. The aforementioned bound is computed by relaxing the original problem into two subproblems, where it matches the optimal solution for the preemptive case of the problem, i.e., $P2|fix_j, r_j, pmtn|C_{\max}$. In what follows, we will explain how the lower bound can be reached.

Let $N = \{1, \ldots, n\}$ be the set of tasks and P_1 and P_2 two processors such that a task j is released at time r_j and has to be processed without preemption during its processing time p_j and, C_j is the completion time of task j while $C_{\max}$ denotes the makespan of the schedule to minimize. A task $j \in N$ is called a $P_1 - task$ (resp. P_2-task) if it is assigned to P_1 (resp. P_2) while it is called P_{12}-task whenever the task j requires simultaneously both P_1 and P_2; that is a processor task. Then, the lower bound can be computed by splitting ST2P into two subproblems, where all bi-processor tasks are divided into two sets of mono-processor tasks each: the first (resp. second) set, noted P_{12}^1-tasks (resp. P_{12}^2-tasks) are separately scheduled on each processor. Thus,

- P_1-tasks and P_{12}^1-tasks should be scheduled on processor P_1.
- P_2-tasks and P_{12}^2-tasks should be scheduled on processor P_2.

Finally, an optimal solution for each subproblem can be provided by processing tasks in a nondecreasing order of their release dates r_j on each processor. Positioning step

by step the tasks assigned to each processor induces an optimal solution for each subproblem, an optimal solution C_1^{opt} for the first subproblem with P_1 and, C_2^{opt} for the second one with P_2. Hence, ST2P's lower bound corresponds to

$$\max(C_1^{\text{opt}}, C_2^{\text{opt}}).$$

Note that the search process used for computing the aforementioned bound is a polynomial-time algorithm with an order time complexity of $O(n \log n)$.

For the rest of the paper, we focus on the main principle of the hybrid method proposed for approximately solving scheduling available tasks on two dedicated processors. The following steps summarize the steps employed by the method:

1. Starting the search process with a greedy solution provided by applying the so-called basic knapsack procedure (cf. Sect. 3.2).
2. Making a series of moves on the current solution for achieving an improved solution (cf. Sect. 3.3).
3. Shaking the order related to some tasks by applying the so-called drop and rebuild strategy (cf. Sect. 3.4).
4. Iterating steps from (1) to (3) till satisfying a stopping criterion.

3.2 A Starting Solution

For an instance of ST2P, any given order of tasks induces a sequence of positions reflecting an assignment of all tasks to the processors. However, using a tailored strategy may often provide a solution with best quality. Indeed, in our study, we propose to build a starting solution by applying a greedy solution procedure, which is based upon the so-called *knapsack rule*. The greedy procedure combines the following two steps: (i) reordering the tasks (items) according to a given criterion and, (ii) selecting step by step a non-affected task (item) and assigning it to a processor (knapsack). The second step is iterated till positioning all tasks (items) on their corresponding processors (knapsacks).

The standard scheduling's greedy procedure may be summarized as follows: let r_j denote the release date of task j and p_j its processing time; then,

1. Compute all ratios related to the processing time per release date, i.e., the value $\frac{p_j}{r_j}$, $j \in N$.
2. Rank all tasks according to the non-increasing order of ratios.

By using the above steps to each task, according to a given order, an initial solution may be built for ST2P. The provided solution is a sequence of tasks assigned to either the first processor, or the second processor, or both processors.

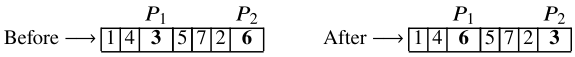


Fig. 1 Permutation between two positions P_1 and P_2 related to two tasks

3.3 An Enhancing Strategy

From a sequence, reflecting a solution, one can build a new sequence, for instance, by forcing the assignment of a subset of tasks to processors and, to complete the provided sub-solution by solving the rest of the subproblem in order to complete the final sequence. Herein, we do it by making moves between tasks, which is equivalent to fix some of tasks (assigned to some processors) and to reassign the rest of the tasks to their corresponding processors(s).

3.3.1 A First Enhancement

The first enhancement used is based upon 2-opt procedure. This procedure is the first local search that is able to improve solutions even if it is based upon small modifications on the solution at hand. Generally, from a given feasible solution, the 2-opt repeatedly makes some moves as long as the quality of the provided solution is improved. Whenever the procedure stagnates around the same bound, then 2-opt is trapped into a local optimum. Herein, the used 2-opt consists in swapping two randomly generated positions from the sequence at hand, where a series of swapping induces the current neighborhood around that solution.

Figure 1 illustrates the permutation used between a pair of positions $(P_1(3), P_2(6))$. By making a swap between two tasks (as illustrated in Fig. 1-right side), the process may induce either a feasible solution or an unfeasible one. Each unfeasible solution can be repaired by applying the steps described below. Let i and j, $i \neq j$, be two assigned tasks (after a permutation), where i is positioned before j. Then the following steps may be applied:

The first-step.

1. According to the position related to task i, move all tasks from the left to the right till removing the infeasibility.
2. According to the position of task j (with its new position), move all tasks from the left to the right till removing all overlapping.

The second-step. Because the swapping operator produces a new sequence, we then apply the knapsack greedy procedure to that order.

Hence, by applying the above steps to the solution at hand, one can observe that a series of solutions can be built and so, these solutions constitute the current 2-opt neighborhood.

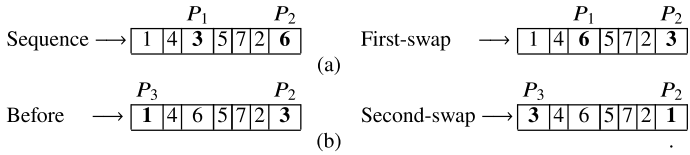


Fig. 2 A second enhancement related to a triplet of positions (two couples of swapping): (**a**) the first couple of positions (P_1, P_2) and, (**b**) the second couple of positions (P_2, P_3)

3.3.2 A Second Enhancement

The second enhancement can be viewed as an extended version of the first one. Indeed, instead of using a couple of positions (two positions), we propose the use of a triplet of positions (three positions). In fact, as observed in Sect. 3.3.1, a current solution may be locally improved by using the first enhancement (2-opt), which is based on small moves. We then propose to introduce a neighbor operator with higher freedom, which can mix two consecutive neighbors around the solution at hand. The main idea of such a strategy is to iterate a series of small moves around the current solution. At a certain internal iteration, to consider an alternative search operator with higher moves and to continue the search by applying small moves.

Each step of the higher move-based operator can be summarized as follows (in what follows, we consider $\hat{S}$ as the solution at hand):

1. Select two random tasks from the solution $\hat{S}$, permute both tasks for forming a new configuration $\hat{S}'$.
2. Randomly select two new tasks from $\hat{S}'$ (different from the already swapped tasks), permute these tasks for forming a new configuration $\hat{S}''$.
3. Call the 2-opt operator on $\hat{S}''$ for improving the quality of the solution and, let $S^\star$ be the new achieved solution (according to $\hat{S}$).

Figure 2 illustrates two successive iterations when applying the second enhancement to a feasible solution.

3.3.3 Avoid Cycling

The goal of both enhancements (Sects. 1 and 2) is to build a series of solutions iteratively reached throughout searching on a number of neighborhoods. In order to avoid cycling (around some already computed solutions), we introduce the so-called tabu list in order to store some moves instead of storing all visited solutions. Because storing these solutions may induce the saturation of the memory-space, the tabu list is then limited to storing some inverse-moves (inverse-swaps) to avoid cycling and stagnation of solutions. Also, the enhancing strategy tries to find solutions throughout a series of neighborhoods provided with both used enhancements (2-opt and 3-opt). Despite the improvements issued from these operators and because both 2-opt and 3-opt use swaps between tasks for iteratively generating a series of neighbors, we

then add a tabu list that stores a list of temporarily inverse-swaps; that are the moves trying to avoid to return to the solutions already visited.

– Drop and Rebuild Operator (DRO)

Input. An instance of ST2P with a solution $\hat{S}$.
Output. A new solution $S^\star$ of ST2P.

1: From the starting solution $\hat{S}$, drop $\beta\%$, $\beta \in]0, 100[$, of tasks according to the current order of the sequence and, let $\hat{S}_1$ be the first partial solution built with the rest of the tasks
2: Solve the reduced problem by using the knapsack procedure (cf., Section 3.2) and let $S^\star$ be the complete achieved solution
3: Improve the current solution $S^\star$ by calling both enhancement phases (cf., Section 3.3), including the already removed tasks
4: **return** $S^\star$

3.4 Exploring the Search Space

Other operators, like local search using diversifications, can also be applied for improving the quality of the solutions. Hifi and Michrafy [8] designed a reactive strategy where two complementary operators are combined for exploring new subspaces. Both dropping and rebuilding operators are incorporated in the search process to favor some fixed items and to optimize the rest of the subproblem with non-fixed variables for completing the solution at hand. In this work, we adapt this strategy to ST2P as follows:

1. Construct a partial solution by removing some tasks from the current solution. Then, a partial solution is reached.
2. Complete the current partial solution (of the first step) by solving the subproblem with the free tasks.

The process consists of dropping a subset of assigned tasks from the current sequence (solution). The dropping phase tries to diversify the search process by degrading the quality of the current solution with the aim of avoiding stagnations and so, to explore new subspaces. The partial solution is built and completed by applying the knapsack procedure, according to the new order associated to the free tasks. Indeed, the diversification strategy can be applied by using the Drop and Rebuild Operator (DRO), which can be summarized in what follows. Let $\hat{S}$ be the current solution, the DRO strategy is then applied in order to reduce the problem, by randomly fixing a subset of already assigned tasks of $\hat{S}$, as follows:

A Hybrid Method (HM)

Input. An instance of SP2P.
Output. A near-optimal solution $S^\star$ with its objective value $C^\star_{\max}$.

1: Set $S^\star = \emptyset$ and $C^\star_{\max} = +\infty$.
2: Call CP for solving the original problem providing the solution $\underline{S}$ with objective value $C_{\max}$.
3: **repeat**
4: **while** (the stopping criterion is not performed) **do**
5: **if** ($C_{\max} < C^\star_{\max}$) **then**
6: set $S^\star = \underline{S}$ and $C^\star_{\max} = C_{\max}$.
7: **end if**
8: **while** (2-opt local iterations is not matched) **do**
9: Call 2-opt using $\underline{S}$'s neighborhood and let $\underline{S}'$ be the neighbor solution with the best objective value $C'_{\max}$.
10: **if** ($C'_{\max} < C^\star_{\max}$) **then**
11: set $S^\star = \underline{S}$ and $C^\star_{\max} = C'_{\max}$.
12: **end if**
13: Update the local iterations and set $\underline{S} = \underline{S}'$.
14: **end while**
15: (i) Call 3-opt using $\underline{S}$'s neighborhood and let $\underline{S}'$ be the neighbor solution with the best objective value $C'_{\max}$.
(ii) Set $\underline{S} = \underline{S}'$ and $C_{\max} = C'_{\max}$.
16: **end while**
17: (i) Apply DRO to the best current solution $S^\star$ and let $\underline{S}$ be the solution reached.
(ii) Reinitialize the 2-opt local iterations.
18: **until** (the global criterion is performed).
19: **return** $S^\star$ with its objective value $C^\star_{\max}$.

3.5 An Overview of the Hybrid Method

The above algorithm describes the main steps of the Hybrid Method (denoted HM). The input of HM is an instance of SP2P and its output is a (near) optimal solution $S^\star$ with its objective value $C^\star_{\max}$. The method starts with an initial solution (line 2) provided by the constructive procedure using the knapsack rule. The method contains three loops: a global loop and two internal loops. The global loop `repeat` from line 3 to line 18 is used for generating a series of solutions, which are enhanced by using both intensification and diversification phases. Its stopping condition is defined according to the number of iterations based on the size of the instance. The first internal loop `repeat` from line 8 to line 14 intensifies the search process by using the first enhancement (2-opt procedure) while the second internal loop (from line 4 to line 16) is applied with the second enhancement (3-opt procedure). The diversification procedure is considered whenever both internal loops stagnate on a local optimum ((i) and (ii) of line 17). Both internal loops are embedded into the global loop `repeat` trying to enhance (and scatter) the new solution generated by

the drop and rebuild operator. The global loop is iterated until either the runtime limit or the number of global iterations is performed. Finally (line 19), the method exits with the best solution found so far $S^\star$ (with objective value $C^\star_{\max}$).

4 Experimental Part

The objective of the experimental part is to evaluate the performance of the Hybrid Method (HM) by comparing its achieved upper bounds (objective values: minimization problem) to the bounds provided by the available methods of the literature. Indeed, its provided results are compared to those achieved by both Kacem and Dammak's [10] Genetic Algorithm (noted GA[1]) and, Manaa and Chu's [11] tight Lower Bound (noted LB), as used in Kacem and Dammak [10].

The behavior of all used methods is evaluated on two different sets of instances: each set is composed of five groups and each group is related to the type of instances considered (as suggested in Manaa and Chu [11]). We note that the proposed method was coded in `C++` and run on an `Intel Pentium Core i7-8550U` with 1.99 GHz and 16 Gb of RAM (all methods were also coded in `C++` and using the same computer).

The instances considered were generated by using Manaa and Chu's generator [11], where five types of instances are considered,[2] according to the number of tasks n to use and the tasks assigned to either P_1, or P_2 and, the bi-processor P_1 and P_2 (noted P_{12}):

- The number of tasks n is fixed to 10 for small-sized instances, to 20 for medium instances, and to 100, 500 and 1000 for large-scale ones: thirty instances are considered for each value.
- The number n_1 (resp. n_2 and n_{12}) of tasks assigned to P_1 (resp. P_2 and P_{12}) is generated following the values displayed in Table 1, where $[x]$ denotes the integral value of x.
- The processing time related to the duration of task j, noted p_j, is randomly generated in the discrete interval $\{1, \ldots, 50\}$.

Table 1 Characteristics of the two sets of instances

Type of task	Type 1	Type 2	Type 3	Type 4	Type 5
n_1	n	n	n	n	$[n/2]$
n_2	$[n/2]$	n	$[n/2]$	n	$[n/2]$
n_{12}	$[n/2]$	$[n/2]$	n	n	n

[1] The code was provided by the first author for generating and testing the behavior of all methods on the same instances.

[2] All tested instances will be made publicly available for other researchers in the domain (https://www.u-picardie.fr/eproad/).

- The release date r_j, related to task j, is randomly generated in the discrete interval $\{1, \ldots, k\}$, with $k = \alpha \times \left(\frac{s_{12}+(s_1+s_2)}{2}\right)$ and $\alpha \in \{0.5; 1; 1.5\}$ (denoting the density of the instance), and s_1 (resp. s_2 and s_{12}) denotes the overall durations related to tasks assigned to P_1 (resp. P_2 and P_{12}).

4.1 Parameter Settings

Often the behavior of a designed heuristic depends on adjustments used on all of its parameters. It means that some adjustments can degrade the quality of the solutions achieved and so, in this part we try to experimentally find the right values to assign. The proposed HM uses three parameters: (i) 2-Opt procedure, (ii) 3-Opt procedure and, (iii) the dropping parameter β used by the diversification strategy.

4.1.1 Effect of the Enhancement Strategy

The enhancement strategy (Sect. 3.4) is composed of two enhancement phases: using the 2-opt procedure and the 3-opt procedure. In order to analyze the behavior of HM when using this strategy, four versions of the method are considered. We first make a preliminary experimental part on the medium-sized instances and, we will generate fix the strategy used for the rest of the instances.

The first version (noted V_1) refers to the versions without enhancements (neither 2-opt nor 3-opt is used), the second version (V_2) incorporates the 2-opt procedure, the third version (noted V_3) refers to the version using the 3-opt procedure while the fourth one (noted V_4) denotes the version using both 2-opt and 3-opt procedures. Note that all tested versions (except V_1) are stochastic algorithms and so, ten trials were considered for each of these versions. The provided results, for the four versions, are reported in Table 2, where the runtime of each version is very small (less than 0.005 s). In order to make a more complete comparison between the four versions, we fix β (the degrading/dropping parameter) to 10% (according to the good behavior of the method when called with this value, as discussed later in Sect. 4.1.2).

Table 3 reports the results provided by the four versions V_1, V_2, V_3 and V_4 and, Manaa and Chu's tight lower bound. Columns 1 and 2 of the table display the instance information: the number n of tasks and the value α representing the density of each instance. Column 3 tallies the lower bound and column 4 the best average upper bounds provided by V_1. Columns 5 and 6 report the average best bounds and the global average bounds of all instances achieved by V_2. Columns 7 and 8 (resp. columns 9 and 10) tally the same values for V_3 (resp. V_4).

Although discussion of the provided results, for the four versions on the medium instances, reported in Table 2, follows:

1. V_1 versus V_2: V_2 outperforms V_1, because V_2 provides a better global average upper bound. Indeed, V_2's average best upper bound is equal to 546.473 while V_1

Table 2 Behavior of HM with and without using 2-opt and/or 3-opt procedures, on instances with $n = 20$ (medium instances)

Instances				V_2		V_3		V_4	
	$n = 20$	LB	V_1	Best	Avg	Best	Avg	Best	Avg
Type 1	$\alpha = 0.5$	354.50	395.80	369.40	378.70	368.00	370.50	362.10	367.80
	$\alpha = 1$	411.60	491.90	459.20	478.00	454.90	464.30	436.40	455.60
	$\alpha = 1.5$	536.30	600.30	569.20	588.10	563.80	569.60	551.80	567.30
Type 2	$\alpha = 0.5$	318.60	377.20	346.30	359.50	338.50	345.90	332.90	342.70
	$\alpha = 1$	471.10	554.70	509.60	535.10	516.80	524.70	494.30	512.70
	$\alpha = 1.5$	703.50	776.70	743.60	762.60	731.40	739.40	721.10	740.60
Type 3	$\alpha = 0.5$	392.80	440.40	412.50	427.80	409.20	413.20	403.20	410.20
	$\alpha = 1$	491.80	585.50	555.30	574.30	535.70	548.40	538.10	551.60
	$\alpha = 1.5$	670.70	775.50	740.40	760.20	717.50	734.70	716.70	733.50
Type 4	$\alpha = 0.5$	443.40	514.60	487.60	500.50	475.90	483.40	468.60	478.60
	$\alpha = 1$	568.50	681.00	635.50	663.90	628.00	637.90	608.50	635.20
	$\alpha = 1.5$	843.30	971.90	918.70	951.40	898.80	911.00	893.20	917.50
Type 5	$\alpha = 0.5$	287.10	329.30	307.80	318.50	306.80	310.40	299.50	307.20
	$\alpha = 1$	395.10	476.80	455.20	466.90	439.30	446.50	436.00	449.10
	$\alpha = 1.5$	643.00	731.00	686.80	711.30	681.90	692.60	661.80	682.50
	Average	502.087	580.173	546.473	565.120	537.767	546.167	528.280	543.473

realizes an average best upper bound of 580.173. Also, V_2's approximation ratio (Ap($A(I)$)=$\frac{A(I)}{\text{LB(I)}}$), where $A(I)$ is the objective value (upper bound) provided by algorithm A for an instance I and, LB denotes the lower bound) equal to 1.15 which is greater than that reached by V_2, which is equal to 1.09. In this case, the 2-opt procedure used is able to enhance the quality of the solutions.

2. V_2 versus V_3: V_3 outperforms V_2. Indeed, V_3 dominates V_1 because its average best global upper bound is equal to 537.767, which is better than that reached by V_2 (546.473). Its approximation ratio Ap(V_3) is now equal to 1.07, which becomes better than Ap(V_3) (that is equal to 1.09). It means that the 3-opt procedure becomes more efficient when used instead of the 2-opt procedure.
3. Combining both 2-opt and 3-opt procedures: V_4 is very competitive when comparing its provided results to those achieved by the best version over the three ones (V_3 instead of V_1 and V_2). Indeed, on the one hand, combining both 2-opt and 3-opt procedures is able to enhance all the average (global) bounds (it becomes now equal to 528.280 when compared to V_3's value − 537.767). On the other hand, Ap(V_4) is equal to 1.05 while Ap(V_3) provides an average value of 1.07.

Figure 3 illustrates the variation of all approximation ratios realized by the four versions: V_1, V_2, V_3 and, V_4 respectively. From the figure, we can observe that combining both 2-opt and 3-opt procedures induces a better approximation ratio, which varies from 1.15 to 1.05.

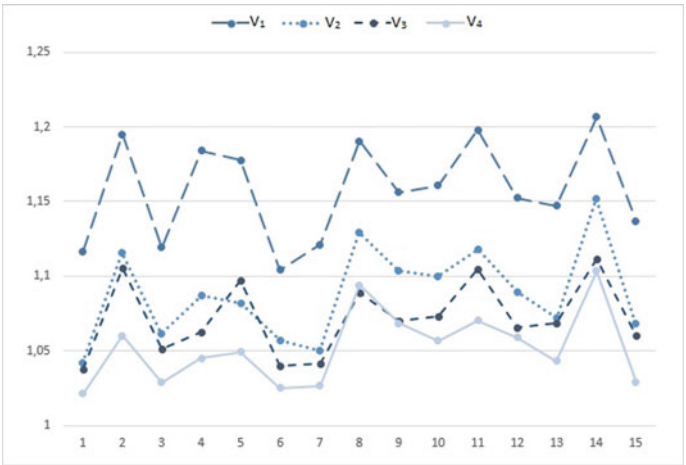


Fig. 3 Representation of the approximation ratios reached by the four versions: Ap(V_1), Ap(V_2), Ap(V_3) and Ap(V_4)

Hence, based on the above analysis, one can expect the good behavior of the proposed HM when combining both the enhancement strategy with the diversification strategy.

4.1.2 Effect of the Diversification Strategy

In order to evaluate the behavior of HM with the parameter β, we introduce a variation on the number of removing tasks in the discrete interval {10, 20, 30, 40, 50}. Table 3 reports the average upper bounds achieved by HM when varying the parameter β for the same instances tested in Sect. 4.1.1. Columns 1 and 2 of the table report the instance information: the number n of tasks and the value α related to the instances represented in Table 3. Columns 3 and 4 show the average bounds of all instances of each group and the average runtime consumed when setting β to 10%. Columns 5 and 6 display the average bounds and the average runtime for $\beta = 20\%$, while columns 7 and 8 (resp. columns 9 and 10 and, columns 11 and 12) report the average bounds and its average runtime when fixing β to 30% (resp. β to 40% and, β to 50%).

In what follows, we comment the results reported in Table 3:

1. HM achieves better average bounds for $\beta = 10\%$ (last line, column 3).
2. When the value of β increases (i.e., β varies from 20 to 50%), the used perturbation is unable to reach better average bounds. We believe that for these largest values of β, HM may explore a largest space and so, the search process is not able to locate good directions for improving the quality of some visited solutions.

Moreover, before fixing the final value of β, for extending the experimental part, we introduce a statistical analysis on the average bounds achieved by the five versions

Table 3 Effect of the dropping parameter β

		Variation of β									
Instances		10%		20%		30%		40%		50%	
	$n = 20$	Av	cpu	Av	cpu	Av	cpu	Av	cpu	Av	cpu
Type 1	$\alpha = 0.5$	359.10	0.0049	359.40	0.0148	359.20	0.0192	359.10	0.0248	360.10	0.0284
	$\alpha = 1$	423.80	0.0063	433.80	0.0225	432.30	0.028	435.00	0.0314	430.40	0.0395
	$\alpha = 1.5$	543.30	0.0054	549.20	0.0157	550.70	0.0187	546.60	0.0238	549.30	0.0318
Type 2	$\alpha = 0.5$	325.00	0.0055	328.70	0.0218	328.90	0.0267	326.80	0.0329	328.00	0.0391
	$\alpha = 1$	486.50	0.0053	493.80	0.0183	491.50	0.0244	488.10	0.0294	488.30	0.0346
	$\alpha = 1.5$	712.90	0.0063	718.50	0.0266	720.70	0.0368	719.30	0.0434	719.50	0.0503
Type 3	$\alpha = 0.5$	397.20	0.0082	401.50	0.0316	399.70	0.0362	402.50	0.0439	399.50	0.0549
	$\alpha = 1$	517.90	0.0064	529.70	0.03	523.90	0.0428	527.70	0.0482	525.70	0.0597
	$\alpha = 1.5$	690.90	0.007	700.00	0.0298	700.60	0.0374	703.90	0.0449	700.40	0.0554
Type 4	$\alpha = 0.5$	457.10	0.0081	463.30	0.0387	461.60	0.047	463.40	0.0608	464.20	0.0827
	$\alpha = 1$	595.80	0.0085	608.50	0.0416	601.70	0.0545	603.70	0.0559	603.40	0.0687
	$\alpha = 1.5$	870.00	0.0124	883.80	0.0523	884.80	0.0638	880.10	0.0703	881.30	0.0747
Type 5	$\alpha = 0.5$	295.80	0.005	297.50	0.016	297.30	0.0208	297.60	0.025	296.40	0.0292
	$\alpha = 1$	419.00	0.003	429.60	0.0075	429.10	0.009	427.40	0.0116	428.00	0.0141
	$\alpha = 1.5$	652.50	0.0054	657.50	0.0146	656.00	0.0202	654.60	0.0235	657.70	0.0276
	Average	**516.45**	0.0065	523.65	0.0255	522.53	0.0324	522.39	0.0380	522.15	0.0460

Table 4 p-values for both *Sign test* and *Wilcoxon rank-test* on the instances with $n = 20$ with the significance level $\theta = 0.05$

	$\mu_1^{(1)}$	$\mu_1^{(2)}$	$\mu_1^{(3)}$	$\mu_1^{(4)}$	μ_2	μ_3	μ_4
p-value (Sign test)	<0.0001	<0.0001	<0.0001	<0.0001	0.941	0.696	0.304
N^+	15	15	14	15	5	7	9
N^-	0	0	0	0	10	8	6
$N^=$	0	0	1	0	0	0	0
p-value (Wilcoxon)	<0.0001	<0.0001	<0.00005	<0.0001	0.938	0.555	0.577

of the algorithm following the variation of β's value. In the comparative study, both the *sign test* and the *Wilcoxon signed-rank test* statistics are considered. We then use the following hypothesis:

$$H_0 : \text{Algo}_{\beta_1} - \text{Algo}_{\beta_2} = \mu, \text{where} \beta_1 \neq \beta_2,$$

to express that algorithm Algo_{β_1} performs better than Algo_{β_2} and, the hypothesis $\overline{H}_0$: algorithm Algo_{β_2} is better than Algo_{β_1} to express the rejection of the hypothesis H_0. Herein, the smallest the average bound and the greater the number of better bounds, the better the corresponding version of Algo.

Table 4 reports, by varying β_1 and β_2 ($\beta_1 \neq \beta_2$), the statistical study on all instances with $n = 20$ by using the *sign rank test* (the detailed results containing the average bounds are those illustrated in Table 3). The different values related to μ represent what follows: $\mu_1^{(k)} = \text{V}_{10\%}$ versus $\text{V}^{(k)}$, $k = 1, \ldots, 4$ and $V^{(k)}$ denotes the versions $\text{V}_{20\%}, \ldots, \text{V}_{50\%}$ and, $\mu_2 = \text{V}_{20\%}$ versus $\text{V}_{30\%}$, $\mu_3 = \text{V}_{30\%}$ versus $\text{V}_{40\%}$ and, $\mu_4 = \text{V}_{40\%}$ versus $\text{V}_{50\%}$. Columns from 1 to 4 display the statistical results of HM when varying β: line 1 displays the p-value corresponding to the sign test, line 2 tallies the number of times that the first version of the algorithm dominates the second version (resp. line 3 reports the number of times that the first version is dominated by the second one and, line 4 shows the number of times that both versions match the same values) and line 5 tallies the p-value related to the rank sign test. From Table 4, we observe what follows:

- For $\mu_1^{(1)}$ the p-value related to the sign test (resp. Wilcoxon rank-test) is smallest to the significance level $\theta = 0.05$, indicating that $\text{V}_{10\%}$ performs better than $\text{V}_{20\%}$ (accepting the hypothesis H_0). Increasing the value of β doesn't improve the quality of the average bounds: p-values related to $\mu_1^{(k)}$, $k \geq 2$, vary from 0.00005 to 0.0001, which means that $\text{V}_{10\%}$ evolves better than the other versions.
- Regarding the comparative study between the four other versions (i.e., $\text{V}_{20\%}, \ldots, \text{V}_{50\%}$), $\text{V}_{30\%}$ performs better than $\text{V}_{20\%}$ according to Wilcoxon's p-value related to μ_2 (equal to 0.938), $\text{V}_{40\%}$ performs better than $\text{V}_{30\%}$ according to Wilcoxon's

p-value related to μ_3 (equal to 0.555) and, $V_{50\%}$ outperforms $V_{40\%}$ regarding μ_4 (the Wilcoxon's p-value is equal to 0.577).

Therefore, because we seek to solutions with high quality (smallest values) and with more best average bounds, we then fix $\beta = 10\%$ for the rest of the experimental part.

4.2 Behavior of HM Versus Available Methods (Set 1)

First, we compare HMs' results to those achieved by GA (Genetic Algorithm−Kacem and Dammak [10]) and Manaa and Chu's [11] tight lower bound (LB).

Table 5 reports LBs and those provided by both GA and HM. Column 3 reports the average lower bound (LB) related to each group of Set 1 (with $n = 10$ and $n = 20$). Column 4 (resp. column 5 and column 6) tallies the best average objective value achieved by GA (resp. the average value over the ten trials and the average runtime over the ten trials). Column 7 (resp. column 8 and column 9) reports the best average objective value achieved by HM (resp. the average values for all trials and the average runtime needed for the same trials). Finally, the last line of the table displays the average values of all values represented in each column (we note that the value in "boldface" (last line of the table) means that the best (average) solution values have been provided by the considered algorithm).

In what follows, we comment results of Table 5:

1. For the small instances (with $n = 10$),
   1. HM outperforms GA although when considering the global average value (the last line of the table of the first block) corresponding to the average solution values over the ten trials of both GA and HM. Indeed, HM achieves an average global value of 634.71 while GA provides an average global value of 768.70. In this case, the gap between both values is closest to 83 units even GA's average runtime remains slightly smaller than that of HM.
   2. HM performs better than GA: its both global average value and global average runtime are better than the one of GA (601.60 instead of 634.71 and, 0.086 s instead of 0.025 s, respectively).
   3. HM outperforms both GA. One can observe that its global average runtime becomes more greater.
2. For the medium-sized instances (with $n = 20$), the same phenomenon can be observed. Indeed, the best HM's average global value (516.45) is better (smallest) than the one achieved by both GA (650.03). HM's global average runtime remains smaller (0.007 s) than that needed by GA (0.237 s).

Figure 4 reports the variation of the approximation ratios provided by both GA and HM. From that figure, HM's approximation ratio is often better than that reached by GA: it varies from 1 to 1.2.

Table 5 Performance of HM versus GA on instances of Set 1: small and medium instances

			GA			HM		
	$n = 10$	LB	UB	Av. UB	T_{GA}	UB	Av. UB	T_{HM}
Type 1	$\alpha = 0.5$	400.90	441.30	467.80	0.068	407.20	407.20	0.012
	$\alpha = 1$	478.70	540.90	575.50	0.065	496.40	496.40	0.011
	$\alpha = 1.5$	789.30	845.50	907.20	0.073	797.90	797.90	0.011
Type 2	$\alpha = 0.5$	402.40	519.10	556.00	0.094	476.60	476.60	0.031
	$\alpha = 1$	651.00	738.10	810.30	0.103	651.10	651.10	0.030
	$\alpha = 1.5$	914.80	1002.70	1067.20	0.093	925.90	925.90	0.033
Type 3	$\alpha = 0.5$	494.90	594.70	640.20	0.104	528.50	528.50	0.028
	$\alpha = 1$	664.00	841.30	895.20	0.094	696.60	696.60	0.029
	$\alpha = 1.5$	924.80	1062.80	1125.40	0.079	936.10	936.10	0.035
Type 4	$\alpha = 0.5$	547.60	690.90	751.30	0.118	582.10	582.10	0.035
	$\alpha = 1$	732.20	959.10	1025.80	0.115	781.70	781.70	0.032
	$\alpha = 1.5$	674.50	767.10	811.60	0.093	681.20	681.20	0.040
Type 5	$\alpha = 0.5$	373.00	444.80	475.30	0.073	397.00	397.00	0.010
	$\alpha = 1$	545.00	655.40	709.40	0.063	560.80	560.80	0.024
	$\alpha = 1.5$	595.50	667.70	712.30	0.061	601.60	601.60	0.011
	Average	612.57	718.09	768.70	0.086	**634.71**	**634.71**	**0.025**
	$n = 20$	LB	UB	Av. UB	T_{GA}	UB	Av. UB	T_{HM}
Type 1	$\alpha = 0.5$	354.50	405.80	427.70	0.162	359.10	361.80	0.005
	$\alpha = 1$	411.60	523.40	567.10	0.165	423.80	439.50	0.006
	$\alpha = 1.5$	536.30	656.10	691.30	0.163	543.30	553.90	0.005
Type 2	$\alpha = 0.5$	318.60	466.00	491.50	0.258	325.00	331.20	0.005
	$\alpha = 1$	471.10	630.60	666.40	0.257	486.50	500.90	0.005
	$\alpha = 1.5$	703.50	838.40	882.30	0.259	712.90	726.00	0.006
Type 3	$\alpha = 0.5$	392.80	519.90	541.50	0.246	397.20	403.30	0.008
	$\alpha = 1$	491.80	690.50	722.40	0.247	517.90	534.20	0.006
	$\alpha = 1.5$	670.70	842.50	884.50	0.245	690.90	709.00	0.007
Type 4	$\alpha = 0.5$	443.40	611.40	640.00	0.357	457.10	466.50	0.008
	$\alpha = 1$	568.50	810.20	853.50	0.3578	595.80	612.80	0.009
	$\alpha = 1.5$	843.30	1059.20	1107.30	0.358	870.00	892.10	0.012
Type 5	$\alpha = 0.5$	287.10	391.30	413.10	0.157	295.80	300.70	0.005
	$\alpha = 1$	395.10	549.30	582.50	0.164	419.00	434.30	0.003
	$\alpha = 1.5$	643.00	755.80	793.60	0.158	652.50	664.50	0.005
	Average	502.09	650.03	684.31	0.237	**516.45**	**528.71**	**0.007**

Table 6 Performance of HM versus GA on instances of Set 2: large-scale instances

			GA			HM		
	$n = 100$	LB	UB	Av	T_{GA}	UB	Av	T_{HM}
Type 1	$\alpha = 0.5$	3 708.60	5 649.80	5 839.70	4.04	3 742.40	3 748.40	0.42
	$\alpha = 1$	4 885.80	7 711.70	7 935.40	4.01	5 167.30	5 261.30	0.40
	$\alpha = 1.5$	7 465.60	10 260.10	10 537.20	4.05	7 508.40	7 595.90	0.41
Type 2	$\alpha = 0.5$	3 793.10	6 753.60	6 940.80	6.50	4 875.80	4 877.10	0.37
	$\alpha = 1$	6 113.90	9 581.90	9 770.80	6.49	6 463.10	6 609.10	0.55
	$\alpha = 1.5$	9 374.60	12 641.50	12 958.60	6.49	9 518.90	9 621.80	0.37
Type 3	$\alpha = 0.5$	4 931.60	7 617.00	7 805.90	6.44	5 012.10	5 026.50	0.60
	$\alpha = 1$	6 181.40	10 312.30	10 570.10	6.41	6 790.50	6 896.20	0.42
	$\alpha = 1.5$	9 019.70	12 908.80	13 181.30	6.42	9 134.50	9 285.40	0.55
Type 4	α=0.5	4 981.60	8 821.40	9 032.20	10.25	6 073.30	6 079.30	0.68
	α=1	7 270.70	12 010.30	12 265.50	9.58	8 098.70	8 214.40	0.67
	α=1.5	10 918.80	15 372.10	15 769.10	9.66	11 120.40	11 259.60	0.77
Type 5	$\alpha = 0.5$	3 855.90	6 204.30	6 394.10	3.97	4 383.80	4 386.20	0.27
	$\alpha = 1$	4 951.80	8 257.70	8 476.00	3.96	5 383.80	5 468.50	0.41
	$\alpha = 1.5$	7 378.20	10 436.40	10 748.60	3.97	7 408.70	7 473.10	0.36
	Average	6322.087	9 635.93	9 881.69	6.15	**6 712.11**	6 **786.85**	**0.48**
	$n = 500$	LB	UB	Av	T_{GA}	UB	Av	T_{HM}
Type 1	$\alpha = 0.5$	18467.40	30480.90	30795.60	168.19	18937.30	19131.90	11.06
	$\alpha = 1$	24451.50	41949.80	42429.60	168.17	28185.70	28520.60	11.16
	$\alpha = 1.5$	36495.10	53447.70	54128.90	90.20	39208.70	39587.50	6.33
Type 2	$\alpha = 0.5$	18669.00	36427.90	36819.70	147.75	24866.90	24980.30	8.93
	$\alpha = 1$	30397.70	50443.20	51160.30	245.19	34720.30	35304.60	13.18
	$\alpha = 1.5$	45492.80	65196.20	65816.10	165.81	48927.10	49443.70	9.77

(continued)

Table 6 (continued)

			GA			HM		
	$n = 100$	LB	UB	Av	T_{GA}	UB	Av	T_{HM}
Type 3	$\alpha = 0.5$	24422.20	40219.20	40675.80	159.43	24992.10	25184.40	7.69
	$\alpha = 1$	30345.30	54303.20	54890.10	141.28	35747.30	36164.10	7.47
	$\alpha = 1.5$	45402.20	68597.60	69516.80	196.38	48813.50	49551.30	9.89
Type 4	$\alpha = 0.5$	24742.10	46583.10	47127.50	289.12	30913.90	31003.40	14.04
	$\alpha = 1$	36931.70	64601.00	65368.60	179.78	42963.30	43754.30	8.88
	$\alpha = 1.5$	54813.00	81707.90	82506.30	347.97	59530.50	60180.70	17.88
Type 5	$\alpha = 0.5$	18501.50	32603.00	33034.20	90.11	21434.90	21478.70	6.13
	$\alpha = 1$	24508.10	44201.30	44705.60	159.29	28729.00	29114.00	9.59
	$\alpha = 1.5$	36516.30	55650.00	56238.50	130.22	39033.10	39597.60	8.12
	Average	31343.727	51094.13	51680.91	178.59	**35133.57**	**35533.14**	**10.01**
	$n = 1000$	LB	UB	Av	T_{GA}	UB	Av	T_{HM}
Type 1	$\alpha = 0.5$	36257.60	61322.00	61809.20	425.55	38297.10	38636.30	19.64
	$\alpha = 1$	48217.50	84189.30	84932.50	426.71	56590.70	57377.30	19.86
	$\alpha = 1.5$	72630.30	108094.00	108963.00	423.89	79063.80	79949.90	19.90
Type 2	$\alpha = 0.5$	37150.80	73880.30	74539.40	698.76	49670.40	49924.50	28.61
	$\alpha = 1$	61249.70	103502.30	104327.50	690.86	71958.80	73364.70	28.44
	$\alpha = 1.5$	91304.70	132736.80	133681.50	691.01	99721.30	101230.50	28.36
Type 3	$\alpha = 0.5$	48673.70	81141.80	81726.00	697.23	50405.50	50853.30	27.20
	$\alpha = 1$	60644.10	110423.60	111298.80	703.04	71806.20	73023.50	27.73
	$\alpha = 1.5$	91361.40	140367.80	141512.60	709.17	101323.80	102708.10	28.02
Type 4	$\alpha = 0.5$	49041.30	94507.80	95303.00	1040.77	61509.00	61748.20	37.39
	$\alpha = 1$	73385.70	130331.40	131307.10	1038.16	88197.70	89749.10	37.40
	$\alpha = 1.5$	109891.60	165420.30	167058.30	1047.41	120859.70	122430.70	37.58
Type 5	$\alpha = 0.5$	36624.50	65940.80	66546.40	445.58	42858.20	43007.30	19.51
	$\alpha = 1$	48729.40	89429.70	90141.50	452.29	58294.70	59218.30	19.45
	$\alpha = 1.5$	73181.70	113430.10	114302.60	452.35	80018.10	81205.10	19.62
	Average	62556.27	103647.87	104496.63	662.85	**71371.67**	**72295.12**	**26.58**

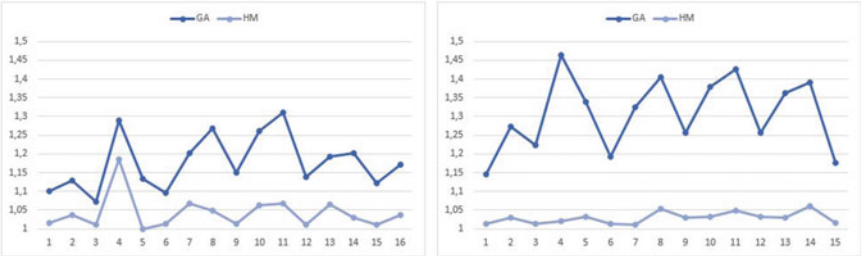


Fig. 4 Variation of the approximation ratios for both GA and HM: (i) the left-side of the figure for $n = 10$ and, (ii) the right-side of the figure for $n = 20$

4.3 Behavior of HM Versus Available Methods (Set 2)

Herein, HM's behavior is analyzed on the instances of `Set 2` which contains three groups representing the large-scale instances: a first group with $n = 100$ tasks, a second group with $n = 500$ tasks and, a third group with $n = 1000$ tasks. Its provided results are also compared to those achieved by the methods available in the literature: GA, and Manaa and Chu's lower bound (LB). Table 6 reports the average bounds provided by HM, GA, and LB on the instances of `Set 2`, where all its information are the same as used in Table 5 (there are three blocks, where each block corresponds to each group of `Set 2`).
In what follows, we comment on the results of Table 6:

1. The first group with $n = 100$: HM's best global average bound (last line of bloc 1) is equal to 6712.113 while those provided by GA is equal to 9635.93. In this case, on the one hand, the gap between both HM and GA is closest to 2923.81. On the other hand, EA(HM) is equal to 1.06 (when comparing its result to that achieved by LB). Unfortunately, HM's average runtime remains also better for instances with $n = 100$ (tasks).
2. The second group with $n = 500$: HM's best global average bound (last line of bloc 2) is equal to 35133.57 while those achieved by the GA is equal to 51094.13. Moreover, on the one hand, one can observe that the gap increases, in this case: it is close to 15960.56 when compared to GA's result. On the other hand, HM consumes smallest runtime when compared to GA's average runtime. We note also that HM's EA is equal to 1.12 while GA's experimental approximation ratio is equal to 1.67.
3. The third group with $n = 1000$: the best global average bound (last line of bloc 3) of HM is equal to 71371.67, which is better (smallest) than those achieved by both GA (103647.87). In this case, HM realizes a gap closest to 32276.2 when compared to GA's results. We note that HM's experimental approximation ratio (EA(HM)) is equal to 1.14 whereas EA(GA) is equal to 1.70. Finally, HM average runtime remains least than that needed by GA for matching the results displayed in the third block (column 8, of Table 6).

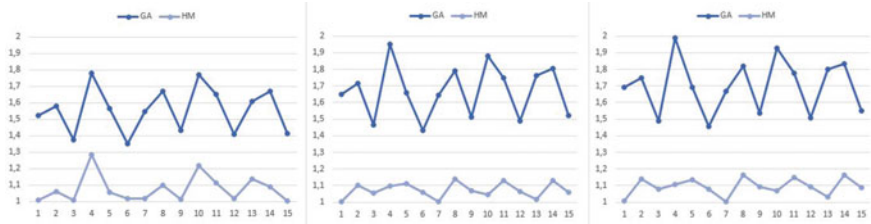


Fig. 5 Variation of the experimental approximation ratios for two tested algorithms GA and HM: (i) on the left-side of the figure for $n = 100$, (ii) on the middle of the figure for $n = 500$ and, (iii) on the right-side for $n = 1000$

Figure 5 shows the variation of the experimental approximation ratios achieved by the three tested methods: GA and, HM. As we can show from that figure, the HM's experimental approximation ratio is often better than those reached by both GA. Indeed, it varies from 1 to 1.29 for $n = 100$, from 1.02 to 1.33 for $n = 500$ and, from 1.04 to 1.34 for $n = 1000$. While for GA it varies from 1.35 to 1.78 for $n = 100$, from 1.43 to 1.95 for $n = 500$ and, from 1.45 to 1.99 for $n = 1000$.

Author contribution statements Mhand Hifi proposed the main idea related to this work. He designed the local hybrid method that combines oscilation and drop/rebuild strategies. Fatma-Zohra Baatout (Phd student supervised by Pr Mhand Hifi) coded the algorithm which is based upon the above idea, and tested it on a set of benchmark instances of the literature. Mhand Hifi investigated the sensitivity analysis, extended the experimental part with a statistical analysis. Méziane Aider is the second supervisor of the PhD student. All authors provided the final version of the paper.

5 Conclusion

In this paper, the problem of scheduling tasks on two dedicated processors is solved with a hybrid method. The proposed method combines four main features. First, an initial solution was built by tailoring a constructive greedy procedure following the knapsack strategy rule. Second, an enhancing strategy was introduced in order to visit a series of local solutions, where a tabu list was also incorporated for avoiding cycling solutions. Third and last, the drop and rebuild operator was considered as a diversification strategy, which was added for exploring some unvisited subspaces. Finally, the performance of the proposed hybrid method was evaluated on a set of benchmark instances containing small, medium and large-scale instances, where its provided results were compared to those achieved by both the genetic algorithm and the tight lower bound tailored for the problem. New results have been provided.

References

1. Aïder, M., Baatout, F.Z., Hifi, M: A reactive search-based algorithm for scheduling multiprocessor tasks on two dedicated processors. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.), Proceedings of the IEEE, Federated Conference on Computer Science and Information Systems, vol. 21, pp. 257–261. ACSIS (2020). https://doi.org/10.15439/2020F134
2. Bianco, L., Blazewicz, J., Dell'Olmo, P., Drozdowski, M.: Preemptive multiprocessor task scheduling with release times and time windows. Ann. Oper. Res. **70**(1), 43–55 (1997). https://doi.org/10.1023/A:1018994726051
3. Blazewicz, J., Dell'Olmo, P., Drozdowski, M., Speranza, M.G.: Scheduling multiprocessor tasks on three dedicated processors. Inf. Process. Lett. **41**, 275–280 (1992). https://doi.org/10.1016/0020-0190(92)90172-R
4. Blazewicz, J., Dell'Olmo, P., Drozdowski.: Scheduling multiprocessor tasks on two parallel processors. RAIRO Oper. Res. **36**, 37–51 (2002). https://doi.org/10.1051/ro:2002004
5. Brucker, P.: Scheduling Algorithms, 4th edn. Springer, Berlin, Heidelberg, New York (2007).ISBN 978-3-540-20524-1
6. Buffet, O., Cucu, L., Idoumghar, L., Schott, R.: Tabu search type algorithms for the multiprocessor scheduling problem. Conf. Artif. Intell. Appl. 674–070 (2010). https://doi.org/10.2316/P.2010.674-070
7. Graham, R.L., Lower, E.L., Lenstra, J.K., Rinnoy, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling theory' a survey. Ann. Discret. Math. **5**, 287–326 (1979). https://doi.org/10.1016/S0167-5060(08)70356-X
8. Hifi, M., Michrafy, M.: A reactive local search-based algorithm for the disjunctively constrained knapsack problem. J. Oper. Res. Soc. **57**(6), 718–726 (2006). https://doi.org/10.1057/palgrave.jors.2602046
9. Hoogeveen, J.A., van de Velde, S.L., Veltman, B.: Complexity of scheduling multiprocessor tasks with prespecified processor allocations. Discret. Appl. Math. **55**, 259–272 (1994). https://doi.org/10.1016/0166-218X(94)90012-4
10. Kacem, A., Dammak, A.: A genetic algorithm to minimize the makespan on two dedicated processors. Control, Decision and Information Technologies (CoDIT), International Conference, pp. 400–404 (2014). https://doi.org/10.1109/CoDIT.2014.6996927
11. Manaa, A., Chu, C.: Scheduling multiprocessor tasks to minimise the makespan on two dedicated processors'. Eur. J. Ind. Eng. **4**(3) (2010). https://doi.org/10.1504/EJIE.2010.033331
12. Thesen, A.: Design and evaluation of Tabu search algorithms for multiprocessor scheduling. J. Heuristics **4**, 141–160 (1998). https://doi.org/10.1023/A:1009625629722

Mathematical Model and Its Optimization to Predict the Parameters of Compressive Strength Test

Adeline Goullieux, Mhand Hifi, and Shohre Sadeghsa

Abstract In this paper, we propose an optimization descent method for predicting compressive material strength estimated parameters for lime and cement as coating substances. We first describe a simple model to express the main parameters used in this study. Second, we describe all equations related to both the amount of water to the amount of coating substance and the ratio of quantity of straw to quantity of coating substance. We then solve the provided model by applying an optimization method that is based upon a variant of the batch-gradient descent method. From a real-experimental case study conducted on the coating of flax straw, its goal is to provide an average estimated error closest to zero for both used materials. Finally, the experimental results shows that the proposed model with its solution procedure is able to predict well the parameters related to all properties.

1 Introduction

We often resort to the use of means of transport resistant to temperature variations, to live in more or less well protected and insulated houses or to work in well soundproofed places, in addition to the characteristics that may be required for a depot. Protecting these moving and immobile assets from the ravages of time and the environment, users have often oriented their choices towards safe structures, such as choosing materials that are inherently strong and often cheaper. The most important mechanical property of a material, such as concrete or an equivalent product, is often related to its compressive strength. The effect of the compressive strength of a mate-

A. Goullieux · M. Hifi (✉) · S. Sadeghsa
EPROAD EA 4669, Université de Picardie Jules Verne, 7, rue du Moulin Neuf, Amiens, France
e-mail: hifi@upicardie.fr

A. Goullieux
e-mail: adeline.goullieux@upicardie.fr

S. Sadeghsa
e-mail: shohre.sadeghsa@upicardie.fr

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_9

rial is a commonly used experiment to measure the different characteristics of the properties of that material.

In this paper, we focus on the effects of the real-experimental case study related to the coating of flax straw before its incorporation into a special material (like cement matrix). During the actual experimental study, the relationship between the amount of water and the amount of coating substance (denoted CS) as well as the relationship between the amount of straw and the amount of CS may vary. Usually, on the one hand, measured characteristics are related to the density, the water absorption capacity and the decrease in water absorption capacity (comparison with untreated straws) of straws. On the other hand, several other properties which can be linked can be taken into account, such as flexural strength, compressive strength, dimensional variation on drying as well as that linked to the extreme dimensional variation of composites. Therefore, the aim of the experiment is to have high mechanical properties and low dimensional variation on the formulated product.

The remainder of the paper is organized as follows. Section 2 reviews some works on the field. Section 3 is composed of three main subsections. First, Sect. 3.1 describes the regression analysis-based statistical approach that can be applied for removing a relationship between both dependent and independent variables. Second, Sect. 3.2 discusses the linear regression-based method used for predicting the main parameters inducing the target material. Third and last, Sect. 3.3 presents both optimization and prediction as a search process. The behavior of the proposed method is computationally analyzed in Sect. 4, where its provided results (estimating parameters) are compared to those achieved by the real-experimentation. Finally, Sect. 5 summarizes the main idea of the study and exposes some future extensions.

2 Background

The mechanical property of a material is often measured by its compressive strength, where in some cases its effect is mainly employed to measure several characteristics related to properties of that material.

As discussed in Siqueira Tango [4], the Compressive Cement Strength Test (namely CCST) was first introduced for analyzing the behavior of the target material, which conventionally takes 28 days for a real-experiment. Of course, two directions may be considered for performing CCST-based experiment:

1. *AST method*: the first way consists in providing a direct real-experimental study, where *Accelerated Strength Testing* approach (namely AST) is applied. In some case, such an approach may be decomposed into several phases (as considered in [1]): (a) using the warm water method; (b) using a boiling water approach; and (c) other approaches.
2. *Soft Computing-Based approach*: the second way consist on predicting target material with its estimated parameters by using optimization and/or learning based methods (cf., de Siqueira Tango [4] and, Snell et al. [8]).

Several methods have been designed for predicting the strength of cement, mortar, or concrete. Indeed, de Siqueira Tango [4] proposed an exploration-based method for estimating the compressive strength of a special material; that is the hydraulic cement product like a Brazilian case study. In the aforementioned study, a function related to resistance according to the time was used to strain the extrapolation of the predicted resistance of the material, where three manners were investigated: (i) taking into account a late age, (ii) taking into account an average age which is composed on two basic resistances and, (iii) low age. The experimental part showed that the proposed method had a good behavior, where it made it possible to obtain satisfactory estimates of the compressive strength of the material studied.

Kabir et al. [7] tackled the prediction of concrete strength by developing a simplified mathematical model. Their model was tailored for the different ages while a single day concrete strength test result is needed. By using some special parameters, the experimental part showed the dependence between parameters' tuning and the quality of the predictions, especially for regulating the concrete strength gaining characteristics regarding the age.

Other popular approaches, like regression methods, have been adapted for estimating (and predicting) several parameters either for mono-criteria or multi-criteria problems (cf., de Siqueira Tango [4] and, Snell et al. [8]). The behavior of materials by itself may depend on both chemical and physical parameters that affect each other in nature. Therefore, detecting and optimizing these materials is not the most efficient method and so, both statistical and analytical models with the use of soft computing-based approaches, including regression analysis seems a promising way to the compressive cement strength prediction problem. The optimization of concrete (and generally composites) concerns selecting values for the constituent parameters. Regarding this direction of research, Baykasoglu et al. [3] proposed a two-step approach for solving the multi-objective optimization of high-strength concretes. In their study, they proposed the use of the regression analysis, the neural networks and the gen expression programming to predict high-strength concretes parameters in the first step. This step established equations related to the concrete characteristic in terms of its components. According to these equations, a multi-objective optimization model has been designed, where it has been solved by using a genetic algorithm; that is the second step of the approach. In Baykasoglu et al. [2] another approach has been proposed for tackling the same problem, where the gene expression programming, te neural networks and the stepwise regression analysis were used as soft computing techniques.

Herein, the problem studied by itself is time consuming and therefore expensive for industries. Because the importance of such a problem, we first propose to simulate he compressive cement strength prediction problem by proposing a series of equations. The equations will return the parameters that needs to be varied in a real-experiment. Results of the equations provided in this study can be replaced with achieved experiment.

Because the proposed method can be viewed as an extended version of the method proposed in Goullieux et al. [6] and, in order to make the paper self-containing, some parts of the aforementioned paper are repeated in what follows.

3 The Compressive Cement Strength: Parameters' Prediction

Practical experiments are conducted in the Laboratory EPROAD, where 123 experiments were conducted on the coating substances including: (i) 60 experiments on *Lime*, (ii) 60 experiments on *Cement* and (iii) three experiments without coating. Because both lime and cement are used, we then considered two different experiments:

1. *Lime's study*. The objective of the first study is to predict the amount of quantity of water to the amount of the quantity of CS and, the ratio of quantity of straw to quantity of CS for lime by having the density and water absorption capacity of straws and flexural strength, compressive strength, density, dimensional variation on drying (Drying Shrikage DS) and Extreme Dimensional Variation (EDV) of composites. This experiment will induce two equations where each of them is used to predict a ratio.
2. *Cement's study*. In addition to the first study, the second experiment will calculate the two ratios related to the cement.

Table 1 illustrates the parameters used in this study, where all parameters are considered in both experiments, i.e., for the *Lime* and the *Cement*.

Therefore, by using all symbols introduced in Table 1,

- On the one hand, for the first experiment the parameters are represented by the set $\{X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$ such that Y_1 denotes the variable to predict for *Lime*.
- On the other hand, for the second equation the parameters are characterized with the set $\{X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$ such that Y_2 is the predicted variable.

We note that for the second experiment, the similar parameters are selected using data of *Cement*.

Table 1 Representation of the used variables

Symbol	Parameter
X_1	Shive water absorbance (%)
X_2	Shive bulk density (kg/m^3)
X_3	Composite Rf (MPa)
X_4	Composite Rc (MPa)
X_5	Composite bulk density (kg/m^3)
X_6	Composite DS (mm/m)
X_7	Composite EDV (mm/m)
Y_1	Shive/Coating substance (S/CS)
Y_2	Water/Coating Substance (W/CS)

3.1 Regression Analysis

Regression analysis is a statistical approach that can be applied for removing a relationship between both dependent and independent variables. It generally may help in modeling some relationship between the used variables, where several types of regressions may be considered, like linear regression, non-linear regression and multiple linear regression. As the literature mentions the preference for the use of linear or multi-linear regressions, we decided to apply the linear regression in this study.

In addition, the use of a linear regression requires the use of one of the gradient descent methods. The gradient descent is an optimization algorithm used for minimizing (or maximizing) the cost (profit) function in various machine learning algorithms. It is basically used for updating the parameters of the considered learning model. As discussed in the literature (cf., Bengio [5] and, Ruder [10]), there are three types of gradient descent-based methods:

- Batch gradient descent method: At each iteration of the search process, all the training samples are processed. However, this version remains interesting when the data size remains small. Therefore, the larger the size of the training examples (data) more the convergence of the method is slower.
- Stochastic gradient descent method: At each iteration of the search process, only one training sample is processed and all parameters are updated. In this case, one can observe that version becomes interesting whenever the data size (training samples) remains small. It induces an acceleration of that version when compared to the previous version, even if it remains heavy when the number of training samples increases.
- Mini-batch gradient descent method: Often this version is faster than the first two versions, even if the manipulation of this approach remains subtle for a good parallelization.

Herein, as mentioned below, data is expensive and collecting the data is time consuming. Therefore, the amount of available data are limited and so, the batch gradient descent calculates the gradient for the whole data set. Because the number of data are limited in this study, we can safely use that version of the descent method.

3.2 Adaptation of the Gradient Descent

In this section, the linear equations are established for predicting the compressive cement strength test (CCST) parameters. Such a model can be described as follows:

$$P(\theta) = \theta_0 + X\theta, \tag{1}$$

where θ_0 refers to a constant value, X denotes the parameters of the model and θ is the coefficient related to all parameters.

3.2.1 Batch Gradient Descent Method

For this version of the gradient descent, the objective function is defined as the average summation of the squared errors of the prediction for the linear regression. Differently stated, that objective function can be written as the following minimisation cost function:

$$C(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(P(\theta)_i - Y_i\right)^2, \tag{2}$$

where C(θ) represents the cost function for each θ, m denotes the number of the training examples while Y is the actual value of the variable; that is predicted using the cost function that should be minimized.

The method tries to minimize the objective function using an iterative procedure, where the coefficients of the parameters of the model are updated as follows:

$$\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} (P(\theta)_i - Y_i) X_i \tag{3}$$

At each iteration j of the gradient descent, θ will be updated to the opposite direction of the gradient of the objective function. The parameter α determines the learning rate. It determines step sizes to reach a local minimum. In other words, at each step we move to the direction of the slope of the objective function towards the bottom of a slope, until a local minimum is reached.

3.2.2 Stochastic Gradient Descent

The difference between the above version of the gradient method lies in the form of the objective function used and the optimization process. As underlined above, it is preferable to use that method whenever the size of data becomes large. Herein, the training process is performed on each training example while all parameters are updated at each iteration. The used cost function is represented as follows:

$$C(\theta, (x^i, y^i)) = \frac{1}{2}\left(P(\theta, x^i) - Y^i\right)^2 \tag{4}$$

and

$$C_{train} = \frac{1}{m} \sum_{i=1}^{m} (C(\theta, (x^i, y^i))), \tag{5}$$

where C(θ) denotes the cost function for each θ, m represents the number of the training examples while Y is the actual value of the variable. Such a value is predicted by using the cost function that should be minimized. Finally, (x^i, y^i) represents features and true value of a training example.

Moreover, the stochastic gradient descent repeatedly applies the following equation:

$$\theta_j := \theta_j - \alpha(P(\theta, x^i) - y^i)x_j^i, \tag{6}$$

where $(P(\theta, x^i) - y^i)x_j^i$ is the derivative of the cost functions such that Eq. (6) will be for each $i = 1, 2, \ldots, m$ and $j = 0, 1, \ldots, n$, where m is the number of examples n the number of features.

3.3 Optimization and Prediction Processes

This section is composed of two parts. First, the *data collection*, which reflects the results induced from the real-experimentation, is presented (where the detailed experiments for both lime and cement are reported in the Appendix). Second and last, the set of data is decomposed into subsets in order to establish correlations between the real experimental values and those to estimate (predict).

3.3.1 Data Collection

Practical experiments are conducted in the EPROAD Laboratory of the University of Picardie Jules Verne, where 123 experiments were conducted on the coating substances including: (i) 60 experiments on *Lime*, (ii) 60 experiments on *Cement* and (iii) three experiments without coating. In the experiment, for each CS, the ratio of quantity of water to quantity of CS is varied in the interval {0.33, 0.5, 0.75, 1} while the ratio of the quantity of straw to quantity of CS is varied in interval {0.5, 0.75 1, 2}.

As illustrated in Table 1, for each CS the measured parameters are shive water absorbance (%), shive bulk density (kg/m^3), composite Rf (MPa), composite Rc (MPa), composite bulk density (kg/m^3), composite DS (mm/m), composite EDV (mm/m).

We recall that the goal of the experiment is to preserve high mechanical properties and low dimensional variation on the formulated coating substances (The experimental data used in this study are summarized in both Tables 7 and 8 of the Appendix: the first table for the lime and the second one for the cement).

3.3.2 The Search Process

Data described and explained in Sect. 3.3.1 are analyzed by applying the following process (for a more detailed comprehensive, the reader may refer to Ng [9]):

Shuffling data. As described in Sect. 3.3.1, data are organized during the experimental test. Thus, we shuffled the data to avoid miscalculations and computational

Table 2 Range of parameters

Coating								
Substance	DataType	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Lime	Average	86.39	201.79	0.93	2.96	741.44	4.94	3.69
Lime	Standard deviation	15.89	49.19	0.61	2.26	118.03	1.42	1.89
Cement	Average	86.78	212.76	1.01	2.62	780.17	5.16	4.44
Cement	Standard deviation	24.10	62.38	0.70	2.04	144.30	1.30	1.07

errors. The shuffling is performed in a way that only the order of the data will be permuted randomly not the parameters of the experiments.

Creating both training and test sets. Data is separated into two groups: the *training set* and the *test set*. Shuffling the data in the previous step is necessary so that there will be no difference in the quality of the data in training and test set. Statistical analysis will be performed in the training set and the quality of the method will be examined in the test set. Note that the size of the test set is setting equal to 20% of the data and the rest (i.e., 80% of data) represents the size of the training set. Of course, limited computational results showed that the used values remains acceptable because of the small number of training samples.

Average and standard deviations. Measured data from CCST has various ranges. Table 2 represents both average and standard deviations for each parameter. Since the range of the values are diversified, one parameter can be 100 times greater than the other one. In this case, scaling the features by mean normalization will drive the gradient descent-based algorithm to converge. Therefore, in order to perform gradient descent on the data, it is advised to mean normalize the data as follows:

$$\frac{X - \mu}{\sigma}, \tag{7}$$

where X denotes the value of the parameter to be normalized, μ and σ represent the average and the standard deviations, respectively. Note that μ and σ are computed once on the training set and then they are applied to normalize the test set.

Randomly initializing θ. The parameter θ is the coefficient of the parameters that is described in Sect. 3.2. Before applying the gradient descent method, θ should be initialized. Herein, it is randomly initialized in the interval [0, 1].

Optimizing the parameter θ. By defining the two parameters (number of iterations and learning rate), the gradient descent algorithm can be applied on the data. One way to verify if the gradient descent algorithm works correctly, is o calculate the cost function and to verify if the value related to the average square of the prediction error as the cost function is decreasing at each iteration.

Using the prediction process. After normalizing test set data, using the average and standard deviations of the training set, predicting the variables may be computed as follows:

$$Y = X\theta, \tag{8}$$

where X denotes the normalized data from the test set while Y is related to the predicted value.

4 Computational Results

The data are analyzed using GNU Octave, version 5.2.0 on a mac OS Catalina with 2.3 GHz Intel Core i5 processor. Parameters tuning of the gradient descent method are considered as follows:

- Learning rate α is setting equal to 0.001, 0.01 and 0.1 and,
- The number of iterations is setting equal to a maximum of 60 000 iterations.

As discussed above, there are 60 data for each experiments; thus, the average runtime is negligible (less than a minute), as underlined in Sect. 3.1.

We note that because two versions of the gradient descent method are considered, we first provide an experimental part with the batch gradient descent method which seems more appropriate for small sized-problems. We then provide a second part in which the stochastique version of the method is applied with the best tuning achieved for the batch version, i.e., the number of iterations used and the best learning rate for α.

Table 3 reports the coefficients of the features when using the optimized θ values. We recall that the equation related to each line of the table is represented as follows:

$$Y_t = \theta_0 + \sum_{k=1}^{m} \theta_k X_k$$

where $m = 7$ and $t = 1$ (resp $t = 2$) for expressing the lime (resp. cement).

Figure 1 (resp. Figs. 2 and 3) illustrates the variation of both real-experimental values related to both Y_1 and Y_1 when considering the prediction for lime −the first graphique− (resp. cement −the second graphique−) while varying the learning rate $\alpha = 0.001$ (resp. $\alpha = 0.01$ and $\alpha = 0.1$).

First, predicted values related to Y_1 for lime seem closest to those extracted from the real-experimentation and those associated to Y_2 have also the same behavior, except for one observation/prediction for which the achieved predicted value is equal to −0.11 while its real-value is equal to 0.50. In this case, this defect can be observed on the three figures while varying α in the interval {0.001; 0.01; 0.1}.

Table 3 Coefficients

Coating substance	Target	α	θ_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Lime	Y_1	0.001	0.89	0.12	−0.27	−0.01	−0.19	0.26	−0.02	0.31
Lime	Y_1	0.01	0.89	−0.02	−0.19	0.12	−0.02	0.31	−0.28	0.28
Lime	Y_1	0.1	0.89	−0.19	−0.28	−0.02	0.31	0.28	0.12	−0.02
Lime	Y_2	0.001	1.06	−0.20	−0.96	−0.23	1.00	−0.05	−0.01	0.11
Lime	Y_2	0.01	1.06	−0.02	−0.06	0.10	−0.23	−1.00	−0.21	1.05
Lime	Y_2	0.1	1.06	−0.06	0.10	−0.23	−0.02	−1.00	1.05	−0.21
Cement	Y_1	0.001	0.85	0.07	0.21	−0.13	0.05	−0.10	−0.12	−0.26
Cement	Y_1	0.01	0.85	0.20	−0.12	0.05	−0.10	0.07	−0.13	−0.26
Cement	Y_1	0.1	0.85	0.07	−0.13	−0.26	0.20	−0.12	−0.10	0.05
Cement	Y_2	0.001	1.14	−0.09	0.51	−0.02	−0.35	0.04	−0.30	0.07
Cement	Y_2	0.01	1.14	−0.09	0.51	0.07	−0.02	−0.35	−0.30	0.03
Cement	Y_2	0.1	1.14	−0.09	0.51	−0.02	−0.30	0.03	0.07	−0.35

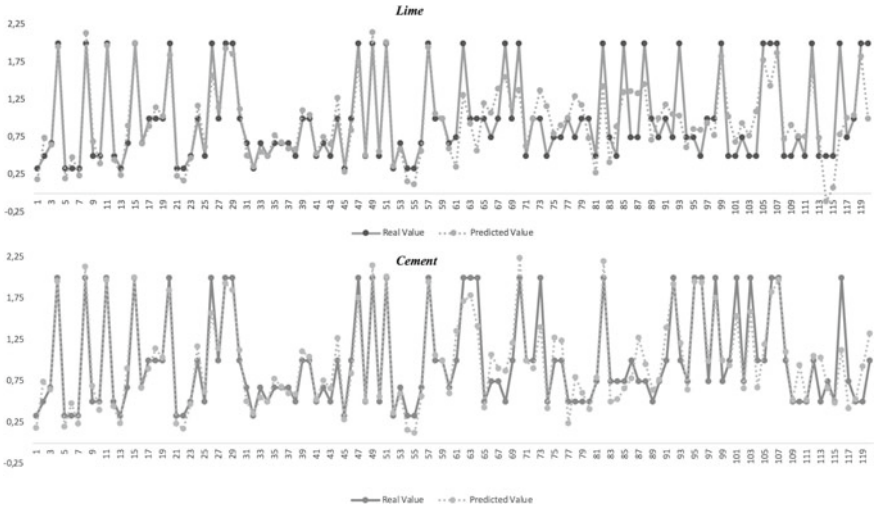


Fig. 1 Real-values versus predicted-values for predicting Y_1 and Y_2 with $\alpha = 0.001$

Second, predicted values related to Y_1 when considering the cement are more interesting. Indeed, in this case, no-defect is observed on the three figures corresponding to the variation of the learning rate α.

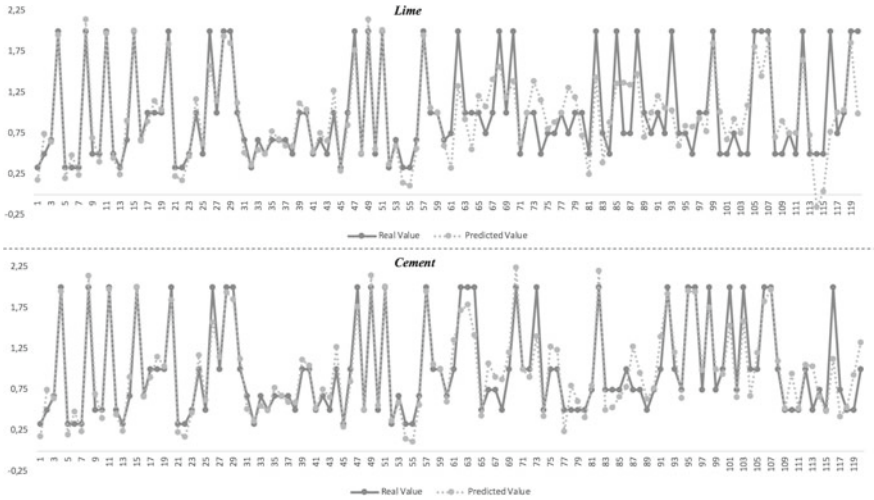


Fig. 2 Real-values versus predicted-values for predicting Y_1 and Y_2 with $\alpha = 0.01$

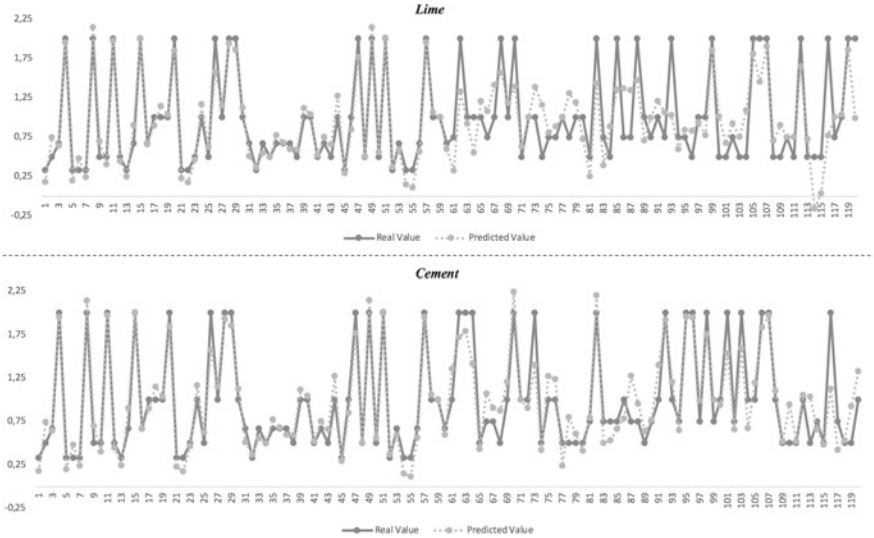


Fig. 3 Real-values versus predicted-values for predicting Y_1 and Y_2 with $\alpha = 0.1$

4.1 Effect of the Number of Iterations

Herein, we evaluate the effect of the number of iterations used on the convergence of the gradient descent method. Indeed, on the one hand, one can observe that increasing the number of iterations, more the cost value decreases, which eventually may converge to a certain value matching a local optimum. On the other hand, as we explained before, by increasing the learning rate, the cost function may also converge faster to a certain local optimum. Whenever increasing the number of iterations it can also induce a stagnation of the value related to the objective function or at least its decreases with a negligible value. Indeed, Fig. 4 illustrates the variation of both the cost value and the number of iterations to predict Y_1 for lime with $\alpha = 0.001, 0.01$ and 0.1. One can observe that the method simultaneously converges to each local optimum, according to the rate α used, by using a closest number of iterations; that is les than the maximum number of iterations fixed. Of course, such a phenomenon can be explained by the fact that training examples is small.

One can observe that the method stagnates with $\alpha = 0.01$, it reduces slowly the cost value with $\alpha = 0.001$ while it seems to converge smoothly towards the final solution with $\alpha = 0.1$. After a number of iterations, the curves related to both $\alpha = 0.01$ and $\alpha = 0.1$ seem equivalent, where they tend to the closest minimal cost objective value.

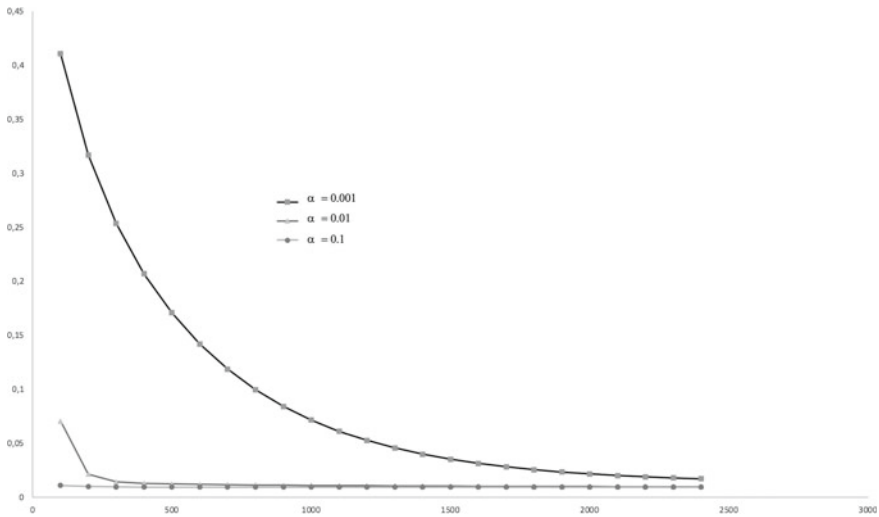


Fig. 4 Behavior of the cost function for predicting Y_1 vs the number of iterations when varying the parameter α

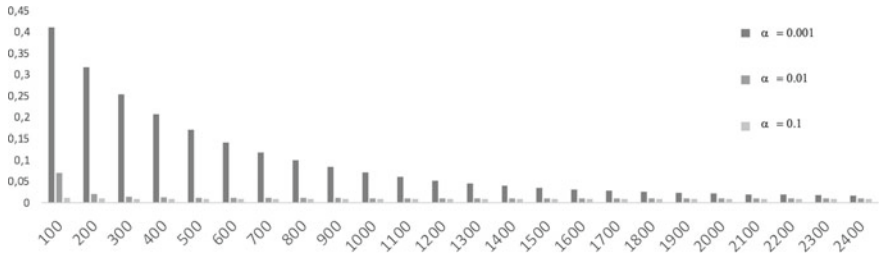


Fig. 5 Variation of the cost function for a maximum of 2400 iteration with varying α to Y_1 (lime)

4.2 Effect of the Learning Rate

We now evaluate the effect of the learning rate on the convergence of the gradient descent method. Generally, more the rate is important more the method converges slowly; that is the principle of the batch gradient descent method. Indeed, Fig. 5 confirms that phenomenon. Indeed, on the one hand, one can observe that the method provides a fast convergence with a highest value of $\alpha = 0.1$ while the smallest value of $\alpha = 0.001$ causes a very slow convergence. On the other hand, with $\alpha = 0.01$ it seems that the method converges towards the final solution as observed for $\alpha = 0.1$.

4.3 Statistical Analysis

In order to evaluate the behavior of the gradient descent method with the parameter α, we consider the same variation as used in previously sections, i.e., α belongs to $\{0.001, 0.01, 0.1\}$. Table 4 reports the absolute errors achieved by the gradient descent method when varying the parameter α in the interval $\{0.001, 0.01, 0.1\}$. Columns 1 and 2 of the table report the instance information: the material and the prediction variable Y_1 and Y_2. Column 3 tallies the real-experimental value of the material and column 4 (resp. column 5 and column 6) displays the absolute errors with $\alpha = 0.001$ (resp. with $\alpha = 0.01$ and with $\alpha = 0.1$).

Moreover, before fixing the final value of α, we introduce a statistical analysis on the average relative errors achieved by the three versions of the method following the variation of the value assigned to α. In the comparative study, both the *sign test* and the *Wilcoxon signed-rank test* statistics are considered. We then use the following hypothesis:

$$H_0 : \mathrm{M}_{\alpha_1} - \mathrm{M}_{\alpha_2} = \mu, \text{ where } \alpha_1 \neq \alpha_2,$$

to express that method M_{α_1} performs better than method M_{α_2} and, the hypothesis $\overline{H}_0$: method M_{α_2} is better than method M_{α_1} to express the rejection of the hypothesis H_0. Herein, the smallest the absolute error and the greater the number of better predictions, the better the corresponding version of the gradient descent method.

Table 4 Absolute errors versus the variation of the learning rate α

			Absolute errors		
CSs	Predict	Real-Value	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
Lime	Y_1	2.00	0.15	0.15	0.15
Lime	Y_1	0.50	0.06	0.06	0.06
Lime	Y_1	2.00	0.01	0.01	0.01
Lime	Y_1	0.33	0.03	0.04	0.04
Lime	Y_1	0.67	0.08	0.08	0.08
Lime	Y_1	0.33	0.17	0.18	0.18
Lime	Y_1	0.33	0.21	0.22	0.22
Lime	Y_1	0.67	0.10	0.11	0.11
Lime	Y_1	2.00	0.05	0.05	0.05
Lime	Y_1	1.00	0.06	0.06	0.06
Lime	Y_1	1.00	0.00	0.01	0.01
Lime	Y_1	0.67	0.07	0.07	0.07
lime	Y_2	0.50	0.41	0.40	0.40
lime	Y_2	0.75	0.02	0.01	0.01
lime	Y_2	0.50	0.26	0.25	0.25
lime	Y_2	2.00	0.38	0.35	0.35
lime	Y_2	0.50	0.24	0.23	0.23
lime	Y_2	0.50	0.61	0.65	0.65
lime	Y_2	0.50	0.42	0.46	0.46
lime	Y_2	2.00	1.21	1.23	1.23
lime	Y_2	0.75	0.26	0.26	0.26
lime	Y_2	1.00	0.04	0.04	0.04
lime	Y_2	2.00	0.17	0.14	0.14
lime	Y_2	2.00	1.00	1.01	1.01
Cement	Y_1	2.00	0.15	0.15	0.15
Cement	Y_1	0.50	0.06	0.06	0.06
Cement	Y_1	2.00	0.01	0.01	0.01
Cement	Y_1	0.33	0.03	0.04	0.04
Cement	Y_1	0.67	0.08	0.08	0.08
Cement	Y_1	0.33	0.17	0.18	0.18
Cement	Y_1	0.33	0.21	0.22	0.22
Cement	Y_1	0.67	0.10	0.11	0.11
Cement	Y_1	2.00	0.05	0.05	0.05
Cement	Y_1	1.00	0.06	0.06	0.06
Cement	Y_1	1.00	0.00	0.01	0.01
Cement	Y_1	0.67	0.07	0.07	0.07
Cement	Y_2	0.50	0.02	0.02	0.02

(continued)

Table 4 (continued)

			Absolute errors		
CSs	Predict	Real-Value	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
Cement	Y_2	0.50	0.44	0.44	0.44
Cement	Y_2	0.50	0.02	0.02	0.02
Cement	Y_2	1.00	0.06	0.06	0.06
Cement	Y_2	0.50	0.54	0.54	0.54
Cement	Y_2	0.75	0.09	0.09	0.09
Cement	Y_2	0.50	0.02	0.02	0.02
Cement	Y_2	2.00	0.87	0.87	0.87
Cement	Y_2	0.75	0.33	0.33	0.33
Cement	Y_2	0.50	0.03	0.03	0.03
Cement	Y_2	0.50	0.43	0.43	0.43
Cement	Y_2	1.00	0.33	0.33	0.33

Table 5 p-values for both *Sign test* and *Wilcoxon rank-test* on all samples and predicted values by varying α: the case of cement

	$\mu^{(1)}$	$\mu^{(2)}$	$\mu^{(3)}$
p-value (Sign test)	0.536	0.536	1
N^+	60	60	56
N^-	60	60	52
p-value (Wilcoxon)	0.463	0.463	1

Table 5 reports, by varying α_1 and α_2 ($\alpha_1 \neq \alpha_2$), the statistical study on the cement material by using the *sign rank test* (the detailed results containing the absolute errors are those illustrated in the second part of Table 4). The different values related to μ represent what follows: $\mu^{(1)} = V_{0.001}$ vs $V_{0.01}$, $\mu^{(2)} = V_{0.001}$ vs $V_{0.1}$ and $\mu^{(3)} = V_{0.01}$ vs $V_{0.1}$. Columns from 2 to 4 display the statistical results of the method when varying α: line 1 displays the p-value corresponding to the sign test, line 2 tallies the number of times that the first version of the method dominates the second version (resp. line 3 reports the number of times that the first version is dominated by the second one) and line 4 tallies the p-value related to the rank sign test. From Table 5, we observe what follows:

- For $\mu^{(1)}$ the p-value related to the sign test (resp. Wilcoxon rank-test) is greatest to the significance level $\theta = 0.05$, indicating that $V_{0.01}$ performs better than $V_{0.001}$ (accepting the hypothesis H_0). Also, column 3 indicates that the version $V_{0.1}$ dominates $V_{0.001}$.
- Regarding the comparative study between the three versions, $V_{0.1}$ performs better than the first two versions according to Wilcoxon's p-value related to $\mu^{(3)}$ (equals to 1).

- In terms of the number of absolute errors matched by the three versions, the first version $V_{0.001}$ provides 50% of best absolute errors, as both other versions, while $V_{0.1}$ achieves 44.33% which is smallest to that provided by $V_{0.01}$.

Therefore, because we seek to solutions with smallest absolute errors and with a greatest number of dominance, we then fix $\alpha = 0.1$ for the rest of the experimental part.

4.4 Behavior of the the Second Version of the Descent Method

Herein, the behavior of the stochastic gradient descent (Noted V2) is analyzed for the target variable Y_1 related to the cement material. The aim of the study is to confirm (or not) the effectiveness of that version while applied to a instance with small-sized. Hence, the method is called by fixing the parameter $\alpha = 0.1$, where the same maximum number of iterations is used. We also note the batch gradient descent method as V1, in what follows.

Table 6 reports the real values corresponding to the values of the real-experiment and those those provided by both V1 and V2. Column 3 reports the predict values achieved by V1 while column 4 tallies those reached by V2. Figure 6 illustrates the variation of the cost (the results detailed in Table 6) reached by both V1 and V2 when compared to the real values. From the figure, on the one hand, we can observe that the curve of V1 is quite close to the curve reflecting the results of the real experiment while that of V2 remains less close. On the other hand, the average squared error (that is the difference between the real value and the predicted value squared) is equal to 0.067 for V1 while that of V2 is equal to 1.6 times larger.

Figure 7 shows the variation of squared errors related to both V1 and V2, where as expressed by both curves, we can remark that V1's curve remains better than that of V2.

Finally, we have already mentioned that the batch version performs generally better than the stochastic version whenever a number of small samples is considered. In order to evaluate the behavior of both versions (a comparative study), applied to the same sample of 60 real data characterizing the cement, we run these versions with the same parameters. Figure 8 illustrates the fluctuations related to both versions V1 (on the left-hand) and V2 (on the right-hand), where V1 seems to converge smoothly towards the final solution while V2 causes a very aggressive oscillation before tending towards a plausible solution. We also note that the average runtime consumed by the bath version is equal to 0.91 seconds while the stochastic version needs 63.54 seconds.

Table 6 Batch method versus stochastic one for predicting the value related to the target variable Y_1: the case of the cement

		Version 1	Version 2
Target	Real	Prediction	Prediction
Y_1	0.67	0.75	1.06
Y_1	0.33	0.41	0.21
Y_1	0.33	0.15	0.18
Y_1	2.00	1.80	1.94
Y_1	0.67	0.76	0.65
Y_1	0.67	1.25	1.27
Y_1	0.50	0.56	0.81
Y_1	0.50	0.80	1.18
Y_1	0.50	0.81	0.55
Y_1	0.67	0.72	0.38
Y_1	2.00	1.73	1.87
Y_1	0.33	0.22	0.48
Y_1	1.00	0.73	0.71
Y_1	0.33	0.26	0.55
Y_1	2.00	1.70	1.77
Y_1	2.00	1.74	1.79
Y_1	0.67	0.68	0.81
Y_1	0.67	0.95	0.99
Y_1	1.00	1.27	1.45
Y_1	0.33	0.24	
Y_1	0.67	0.57	0.09
Y_1	1.00	1.03	1.11
Y_1	2.00	1.55	1.60
Y_1	1.00	1.09	1.33
Y_1	0.33	0.05	−0.02
Y_1	0.67	0.93	0.95
Y_1	2.00	1.75	1.57
Y_1	1.00	1.22	1.09
Y_1	0.67	0.53	0.38
Y_1	0.67	0.98	1.38
Y_1	0.50	0.46	0.22
Y_1	0.50	0.57	0.38
Y_1	2.00	1.53	1.46

(continued)

Table 6 (continued)

		Version 1	Version 2
Target	Real	Prediction	Prediction
Y_1	0.33	0.40	0.51
Y_1	0.67	0.67	0.69
Y_1	0.50	0.63	0.83
Y_1	1.00	1.36	1.26
Y_1	0.50	0.67	0.77
Y_1	0.33	0.14	0.04
Y_1	1.00	1.39	1.20
Y_1	0.33	0.21	0.17
Y_1	2.00	1.75	1.97
Y_1	1.00	0.95	1.30
Y_1	0.67	0.81	0.76
Y_1	0.50	0.11	0.51
Y_1	0.50	0.34	−0.26
Y_1	0.50	0.56	0.50
Y_1	2.00	1.56	1.40
Y_1	0.50	0.72	0.86
Y_1	0.33	0.40	0.31
Y_1	1.00	1.19	1.19
Y_1	0.50	0.78	0.80
Y_1	0.33	0.29	0.32
Y_1	1.00	1.08	1.08
Y_1	2.00	1.51	1.68
Y_1	0.33	−0.01	0.25
Y_1	2.00	1.44	1.36
Y_1	2.00	1.57	1.38
Y_1	1.00	1.40	1.39

5 Conclusion

In this paper we studied effects of a real-experimental case related to the coating of flax straw before its incorporation into a cement matrix. A simple model was proposed for tackling the problem, where a batch gradient descent method was employed for predicting the parameters related to the compressive cement strength. Such a problem needs 28 days for achieving a final material while the proposed method converge to good approximate solutions by consuming less than one minute.

As the studied problem is related to a practical situation, there are plenty possibilities for further investigation involving efficient methods able to predict the parameters. First, neural networks can be applied to predict some parameters and so, in

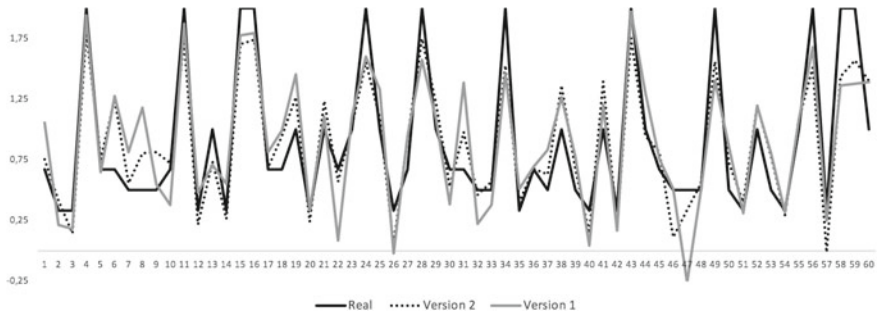


Fig. 6 Cost fluctuation of both V1 and V2 on the training samples regarding the cement material: the case of the target variable Y_1

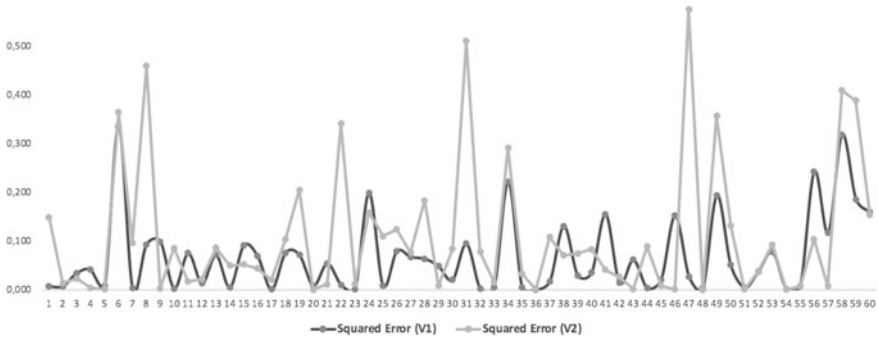


Fig. 7 Variation of the squared error of both costs (compared to the real value) provided by both V1 and V2: the case of the target variable Y_1 (cement)

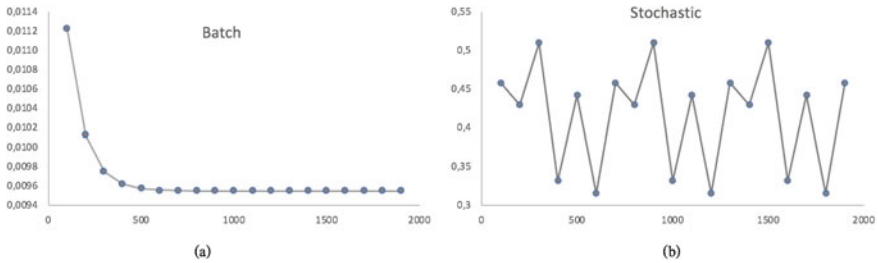


Fig. 8 Variation of the cost when using **a** V1 and **b** V2: the case of the target variable Y_1 (training set) for cement

some cases they can be used for providing tight estimations. Second, the hybridization between neuronal networks and operational research technics can also be envisaged for achieving better predictions of the parameters, especially when multi-objective functions, like predicting parameters and minimizing the overall price of the raw material required, to design the target material.

Appendix

Herein, the real data for each experiment related to both lime and cement are represented in Tables 7 and 8, respectively

Table 7 Experimental data for each experiment of Lime

Coating									
Substance	Y_1	Y_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Lime	0.33	0.50	71.031	240	1.357	6.65	812.227	5.454	2.756
Lime	0.33	0.50	70.181	266.1	1.076	5.994	810.469	5.302	2.69
Lime	0.33	0.50	74.639	264.15	0.749	6.256	836.094	2.958	2.656
Lime	0.33	0.75	84.603	260.95	2.083	8.138	1025.339	2.916	2.554
Lime	0.33	0.75	86.271	256.8	1.638	8.981	1047.93	2.645	1.87
Lime	0.33	0.75	81.717	265.55	1.872	9.556	1011.992	3.894	2.295
Lime	0.33	1.00	68.983	291.95	2.012	4.738	911.354	4.606	1.838
Lime	0.33	1.00	70.052	296.15	1.732	4.8	910.859	4.41	1.903
Lime	0.33	1.00	67.112	297.3	1.638	5.1	911.875	5.42	2.095
Lime	0.33	2.00	71.285	281.3	0.281	0.631	705.039	3.238	1.722
Lime	0.33	2.00	71.633	292.6	0.257	0.694	725.938	3.77	1.31
Lime	0.33	2.00	73.438	298.5	0.304	0.675	691.328	6.492	1.431
Lime	0.50	0.50	77.848	221.55	0.491	1.788	643.555	5.454	2.798
Lime	0.50	0.50	81.236	215.4	0.515	1.575	639.688	5.302	3.28
Lime	0.50	0.50	69.654	215.35	0.655	1.95	651.172	2.958	2.898
Lime	0.50	0.75	73.469	216.75	0.819	2.813	685.195	3.545	2.848
Lime	0.50	0.75	68.654	217.65	0.796	2.769	671.016	4.944	3.227
Lime	0.50	0.75	78.013	227.65	0.866	2.781	705.352	3.683	2.972
Lime	0.50	1.00	73.763	211.05	1.544	6.644	914.18	3.211	2.075
Lime	0.50	1.00	72.059	226.55	1.381	6.269	897.266	4.122	3.429
Lime	0.50	1.00	75.692	218.45	1.661	6.244	930.742	2.403	3.047
Lime	0.50	2.00	71.701	216.1	0.819	1.938	734.883	6.48	3.355
Lime	0.50	2.00	70.274	220.3	0.796	2.25	731.172	4.063	1.495
Lime	0.50	2.00	76.993	219.65	0.889	1.963	730.586	7.784	2.686
Lime	0.67	0.50	86.806	215.35	0.959	2.55	696.172	5.581	3.17
Lime	0.67	0.50	92.871	207.3	1.076	1.925	700.234	5.774	3.565
Lime	0.67	0.50	70.475	213.3	0.983	2.2	680.547	5.56	3.323
Lime	0.67	0.75	94.567	192.05	0.866	3	702.383	3.784	2.734
Lime	0.67	0.75	90.641	195.3	1.17	2.669	693.047	6.846	2.706
Lime	0.67	0.75	98.472	194.7	1.006	3.45	716.25	3.327	2.85
Lime	0.67	1.00	86.515	207.5	1.685	4.044	779.18	5.405	2.814
Lime	0.67	1.00	81.803	213.2	1.685	3.631	760.898	3.29	2.63
Lime	0.67	1.00	47.933	205	1.802	3.25	795.273	2.976	2.697
Lime	0.67	2.00	83.439	203.1	2.574	5	836.198	3.542	2.132

(continued)

Table 7 (continued)

Coating

Substance	Y_1	Y_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Lime	0.67	2.00	82.937	197.15	2.317	4.75	792.344	2.695	2.405
Lime	0.67	2.00	80.99	190.65	2.34	5.25	839.922	3.263	2.311
Lime	1.00	0.50	92.349	183.3	0.608	1.581	651.263	4.094	4.788
Lime	1.00	0.50	100.362	177.05	0.562	1.419	661.016	6.6	4.876
Lime	1.00	0.50	93.248	191.85	0.585	1.381	658.477	8.173	4.371
Lime	1.00	0.75	88.156	152.65	0.468	1.556	676.641	5.421	3.872
Lime	1.00	0.75	87.234	162.25	0.562	1.756	690.977	5.019	4.056
Lime	1.00	0.75	90.253	166.85	0.608	1.963	672.031	5.425	4.35
Lime	1.00	1.00	90.37	174.6	0.842	2.438	686.953	5.101	3.318
Lime	1.00	1.00	82.245	169.6	0.772	1.956	693.633	4.355	3.862
Lime	1.00	1.00	99.006	166.95	0.749	2.588	694.297	5.636	3.761
Lime	1.00	2.00	88.778	166.85	0.749	2.675	779.206	4.045	2.589
Lime	1.00	2.00	81.107	176.6	0.913	2.563	742.422	4.192	2.724
Lime	1.00	2.00	98.734	177.25	0.936	2.569	782.109	5.479	3.101
Lime	2.00	0.50	113.138	143.15	0.187	0.706	602.448	6.125	8.108
Lime	2.00	0.50	136.398	137	0.164	1.075	616.055	7.61	7.911
Lime	2.00	0.50	93.137	139.85	0.211	0.713	610.234	6.318	8.648
Lime	2.00	0.75	112.381	130.75	0.187	0.75	625.43	6.968	4.995
Lime	2.00	0.75	93.471	137	0.257	0.719	625.508	5.617	8.477
Lime	2.00	0.75	118.891	136.85	0.14	1.088	629.805	6.367	7.221
Lime	2.00	1.00	109.414	131.75	0.164	0.838	591.771	6.437	7.278
Lime	2.00	1.00	118.127	129.7	0.281	1.069	583.633	6.021	7.732
Lime	2.00	1.00	101.876	131.35	0.211	0.956	605.313	6.385	7.577
Lime	2.00	2.00	114.51	130.55	0.281	0.938	693.854	5.802	5.108
Lime	2.00	2.00	129.682	135.25	0.304	1.131	686.094	6.557	7.108
Lime	2.00	2.00	117.383	131.5	0.257	1.106	669.883	5.179	6.436

Table 8 Experimental data for each experiment of Cement

Coating

Substance	Y_1	Y_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Cement	0.33	0.50	59.946	320.15	0.468	1.25	811.406	5.261	3.708
Cement	0.33	0.50	69.067	304.45	0.679	1.313	829.258	6.503	3.827
Cement	0.33	0.50	54.545	330.2	0.491	1.313	791.328	6.503	3.731
Cement	0.33	0.75	55.089	320.35	1.076	2.1	905.586	3.898	3.983
Cement	0.33	0.75	54.961	311.05	0.702	2	909.258	5.947	3.503
Cement	0.33	0.75	80.284	335.9	0.936	1.813	906.719	5.17	3.152
Cement	0.33	1.00	60.361	340.75	1.006	3.519	924.961	6.171	3.401
Cement	0.33	1.00	70.052	320.55	0.796	2.875	974.297	3.939	3.144
Cement	0.33	1.00	54.042	323.15	1.498	2.55	901.367	4.916	3.234
Cement	0.33	2.00	79.858	292.25	1.568	6.563	995.234	3.751	3.184
Cement	0.33	2.00	50.47	294.75	1.615	6.556	974.961	4.435	3.452
Cement	0.33	2.00	85.714	315.25	1.568	5.913	965.195	5.555	3.326
Cement	0.50	0.50	73.815	243.9	0.702	1.619	760.586	5.309	3.753
Cement	0.50	0.50	82.759	224.45	0.515	1.6	769.922	4.186	4.227
Cement	0.50	0.50	72.098	232.1	0.796	1.538	754.219	5.63	3.598
Cement	0.50	0.75	66.928	235.5	0.679	1.669	837.852	4.462	3.49
Cement	0.50	0.75	67.366	233.85	0.585	1.744	838.359	5.126	3.612
Cement	0.50	0.75	63.943	251.35	0.749	1.8	792.5	4.584	3.679
Cement	0.50	1.00	66.03	234.35	1.404	2.7	844.414	5.328	3.634
Cement	0.50	1.00	61.719	224.3	1.31	2.006	838.359	4.595	3.487
Cement	0.50	1.00	91.884	244.4	1.264	1.913	837.852	5.693	3.176
Cement	0.50	2.00	87.66	263.35	2.808	6.713	1055.547	4.4	3.588
Cement	0.50	2.00	89.893	249.4	1.895	6.506	1099.492	3.765	3.739
Cement	0.50	2.00	67.757	254.4	2.317	6.194	1073.008	0	3.208
Cement	0.67	0.50	81.06	211.95	0.632	1.75	725.82	6.845	6.166
Cement	0.67	0.50	83.977	202.2	0.702	1.35	748.945	5.143	6.419
Cement	0.67	0.50	91.859	190.05	0.866	1.413	737.305	5.1	5.023
Cement	0.67	0.75	65.435	206.2	1.544	2.606	760	5.334	5.468
Cement	0.67	0.75	79.749	191.35	0.842	3.463	743.047	5.862	5.065
Cement	0.67	0.75	128.798	195.4	0.913	2.175	743.828	5.068	5.042
Cement	0.67	1.00	67.597	198.7	1.24	3.225	726.602	3.808	4.254
Cement	0.67	1.00	86.014	207.9	1.568	2.875	733.672	4.879	4.55
Cement	0.67	1.00	87.159	203.4	0.796	2.813	736.172	3.811	3.946
Cement	0.67	2.00	80	184.95	3.276	6.444	912.773	3.347	3.924
Cement	0.67	2.00	82.085	211	1.919	7.406	937.266	5.407	4.406

(continued)

Table 8 (continued)

Coating

Substance	Y_1	Y_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Cement	0.67	2.00	85.475	200.95	2.153	5.725	898.477	2.858	4.041
Cement	1.00	0.50	102.193	171.65	0.421	0.85	661.25	7.05	5.788
Cement	1.00	0.50	108.021	159.6	0.374	0.788	633.281	5.581	6.014
Cement	1.00	0.50	81.448	161.55	0.304	0.9	612.227	7.086	6.36
Cement	1.00	0.75	87.454	167.1	0.632	1.369	603.789	6.253	6.566
Cement	1.00	0.75	87.234	163.4	0.445	0.969	634.336	5.371	5.708
Cement	1.00	0.75	90.253	167	0.398	1.25	606.172	7.599	5.73
Cement	1.00	1.00	106.072	163.15	0.585	1.438	684.57	5.877	4.161
Cement	1.00	1.00	85.666	160.2	0.515	1.038	686.211	6.021	4.535
Cement	1.00	1.00	82.213	168	0.515	1.519	686.445	4.515	5.808
Cement	1.00	2.00	112.333	152.3	1.31	2.863	753.047	4.71	3.886
Cement	1.00	2.00	113.636	154.3	1.147	2.206	724.414	5.568	4.174
Cement	1.00	2.00	46.812	168.8	1.193	2.575	769.063	4.969	3.871
Cement	2.00	0.50	112.762	140.65	0.257	0.388	596.758	7.128	6.46
Cement	2.00	0.50	110.828	138.9	0.164	0.313	627.344	7.558	5.191
Cement	2.00	0.50	126.966	141.25	0.187	0.488	592.93	7.655	6.761
Cement	2.00	0.75	123.774	135.75	0.351	0.575	606.836	6.769	6.381
Cement	2.00	0.75	127.657	134.6	0.257	0.475	575.313	6.545	6.763
Cement	2.00	0.75	132.538	138.45	0.164	0.463	607.109	7.525	5.473
Cement	2.00	1.00	119.208	133.05	0.281	0.563	612.227	5.988	4.385
Cement	2.00	1.00	119.938	130.6	0.234	0.506	583.945	6.35	4.583
Cement	2.00	1.00	131.653	134.25	0.234	0.5	578.242	5.94	5.347
Cement	2.00	2.00	124.673	130.6	0.351	0.7	574.57	5.734	4.185
Cement	2.00	2.00	120.671	135.3	0.421	1.006	599.063	5.164	3.695
Cement	2.00	2.00	97.852	133.95	0.328	0.794	556.094	6.061	4.909

References

1. ASTM C68499 (2003): Standard test method for making, accelerated curing, and testing concrete compression test specimens. ASTM International, West Conshohocken, PA (1999). https://doi.org/10.1520/C0684-99R03
2. Baykasoğlu, A., Dereli, T., Tanş, S.: Prediction of cement strength using soft computing techniques. Cement Concret Res. **34**(11), 2083–2090 (2004)
3. Baykasoglu, A., Oztas, A., Ozbay, E.: Prediction and multi-objective optimization of high-strength concrete parameters via soft computing approaches. Expert Syst. Appl. **36**(3), 6145–6155 (2009)
4. de Siqueira Tango, C.E.: An extrapolation method for compressive strength prediction of hydraulic cement products. Cem. Concr. Res. **28**(7), 969–983 (1998)
5. Bengio, Y., Boulanger-Lewandowski, N., Pascanu, R.: Advances in optimizing recurrent networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8624–8628. IEEE, May 2013

6. Goullieux, A., Hifi, M., Sadeghsa, S. (2020). An iterative descent method for predicting the compressive cement strength estimated parameters. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.), Communication Papers of the 2020 Federated Conference on Computer Science and Information Systems. ACSIS, vol. 23, pp. 7–12
7. Kabir, A., Hasan, M., Khasro Miah, M.D.: Strength prediction model for concrete. ACEE Int. J. Civil Environ. Eng. **2**(1), 14–19 (2013)
8. Snell, L.M., Roekel, Van, J.,& Wallace, N. D. : Predicting early concrete strength. Concr. Int. **11**(12), 43–47 (1989)
9. Ng, A.: Machine Learning. Stanford University (2018). [lnea]. https://www.coursera.org/learn/machine-learning. (Último acceso: 14 Apr. 2017)
10. Ruder, S.: An Overview of Gradient Descent Optimization Algorithms (2016). arXiv:1609.04747

Optimal Tree of a Complete Weighted Graph

Seyed Soheil Hosseini, Nick Wormald, and Tianhai Tian

Abstract Take a complete weighted graph K_n. We want to find a weighted tree on which the path length between any two vertices is as close as possible to their corresponding distance in K_n. The optimality criterion we have chosen for this problem is the residual sum of squares (RSS). To find the optimal tree, we face two challenges: finding the optimal edge weights on a given tree and finding the optimal tree structure. For the former, we make use of the Cholesky decomposition, and for the latter, we use two metaheuristics: Simulated Annealing (SA) and Iterated Local Search (ILS). Our results suggest that SA and ILS both perform well at finding the optimal tree structure when the dispersion of distances in the complete graph is large. However, when the dispersion of distances is small, ILS has a much better performance.

Keywords Optimal tree · Simulated annealing (SA) · Iterated local search (ILS)

1 Introduction

We want to find an edge-weighted tree that best estimates the complete weighted graph of distances between vertices such that the discrepancy between the path length between any two vertices in the tree, and their distance in the complete graph, is minimized. To this end, we use the residual sum of squares (RSS) as it is a typical optimality criterion for these types of problems. We call the resulting tree, the residual

S. Soheil Hosseini (✉) · N. Wormald · T. Tian
School of Mathematics, Monash University, Victoria 3800, Australia
e-mail: soheil.hosseini@monash.edu

N. Wormald
e-mail: nick.wormald@monash.edu

T. Tian
e-mail: tianhai.tian@monash.edu

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_10

sum of squares optimal tree (RSSOT). The underlying idea for this problem originates from three areas: stock-correlation networks, phylogenetic trees [4] and t-spanners [8] in graph theory. In the first two areas, several algorithms have been proposed to build a network based on the complete weighted graph of distances between stocks [1, 2, 5–7, 9–11] or species information [3]. In the third area, the problem is similar to estimating the t-spanner tree of K_n.

We take an approach similar to some investigations in phylogenetic trees [3], but we have a different treatment of a basic improvement step used in local search heuristics. Also, in contrast to phylogenetic trees, we consider distances between all vertices of the tree, not just leaves. We investigate two metaheuristics—Simulated Annealing (SA) and Iterated Local Search (ILS)—for this problem.

In Sect. 2, we discuss how to optimise edge weights of a given tree. In Sect. 3, we use the aforementioned metaheuristics to optimise the tree structure—find RSSOT—and ultimately, Sects. 4 and 5 include our results and conclusion.

2 Sub-Problem: Tree Weight Optimisation

For the complete weighted graph $K_n = (V, \mathbf{E}, d)$, we want to come up with a weighted spanning tree $T = (V, E, w)$ where $E \subset \mathbf{E}$ such that the path length between any two vertices on the tree best estimates the distance between them in K_n. To be precise, we want to minimise the RSS between path lengths in T and their corresponding edge distances in K_n such that

$$RSS\,(T, K_n) = \sum_{\substack{m,k \\ m<k}} \left(S\left(P_{m,k}\right) - d_{mk}\right)^2. \tag{1}$$

In the equation above, $P_{m,k}$ denotes the path with the start vertex of v_m and the end vertex of v_k in T. Also, $S(P_{m,k})$ denotes the sum of edge weights on this path. For example, for the path $P_{m,k} = (e_{ma}, e_{ab}, e_{bc}, \ldots, e_{dk})$, $S(P_{m,k}) = w_{ma} + w_{ab} + w_{bc} + \ldots + w_{dk}$. Thus, Eq. 1 can be reformulated as

$$RSS(T, K_n) = \sum_{\substack{m,k \\ m<k}} \left(\sum_{\substack{i,j \\ e_{ij} \in P_{m,k}}} w_{ij} - d_{mk} \right)^2. \tag{2}$$

In order to find the edge weights for a given spanning tree, we take the derivative of RSS with respect to the w_{ij}'s, so that $\frac{\partial RSS}{\partial w_{ij}} = 0$. It gives us

$$\frac{\partial RSS}{\partial w_{ij}} = 2\left(\sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}}\left(w_{ij} + \sum_{\substack{e_{rs}\in P_{m,k}\\ e_{rs}\neq e_{ij}}} w_{rs}\right) - \sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}} d_{mk}\right) = 0 \qquad \forall e_{ij} \in E. \tag{3}$$

The equation above can be written as

$$\frac{\partial RSS}{\partial w_{ij}} = 2\left(\alpha_{ij} w_{ij} + \sum_{e_{rs}\neq e_{ij}} \beta_{rsij} w_{rs} - \sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}} d_{mk}\right) = 0 \qquad \forall e_{ij} \in E \tag{4}$$

where α_{ij} denotes the number of paths that edge e_{ij} is on, and β_{rsij} denotes the number of paths on which both edges e_{ij} and e_{rs} are. The reason being each term $(.)^2$ in RSS denotes the square error between a path length in T and its corresponding edge distance in K_n. We have $\binom{n}{2}$—equal to the number of paths between each two vertices in T—of these terms. Taking the derivative with respect to a w_{ij}, we are considering only the terms $(.)^2$ that include the edge e_{ij} which correspond to the paths that include edge e_{ij}. From Eq. 3, we have the following $n-1$ equations

$$\sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}}\left(w_{ij} + \sum_{\substack{e_{rs}\in P_{m,k}\\ e_{rs}\neq e_{ij}}} w_{rs}\right) = \sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}} d_{mk} \qquad \forall e_{ij} \in E \tag{5}$$

or as with Eq. 4 the above can be written as

$$\frac{\partial RSS}{\partial w_{ij}} = \alpha_{ij} w_{ij} + \sum_{e_{rs}\neq e_{ij}} \beta_{rsij} w_{rs} = \sum_{\substack{m,k:\, e_{ij}\in P_{m,k}\\ m<k}} d_{mk} \qquad \forall e_{ij} \in E. \tag{6}$$

The above linear system can be expressed in matrix form as $A\mathbf{w} = \mathbf{d}$, and the entries of matrix A are

$$a_{ij} = \begin{cases} \text{\# of paths including the edge corresponding to the i}-th \text{ entry of vector } \mathbf{w} & i = j \\ \text{\# of paths including the edges corresponding to the i}-th \text{ and the } j-th \text{ entries of vector } \mathbf{w} & i \neq j \end{cases}. \tag{7}$$

Let us go through the following example to make it more clear in Fig. 1.

For the tree, the system of equations is as below.

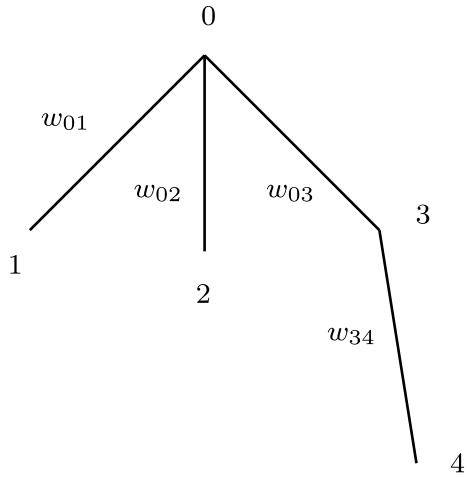


Fig. 1 An example of what matrix A and vector $\mathbf{d}$ are on this tree

$$\underbrace{\begin{bmatrix} 4 & 1 & 2 & 1 \\ 1 & 4 & 2 & 1 \\ 2 & 2 & 6 & 3 \\ 1 & 1 & 3 & 4 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} w_{01} \\ w_{02} \\ w_{03} \\ w_{34} \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} d_{01} + d_{12} + d_{13} + d_{14} \\ d_{02} + d_{12} + d_{23} + d_{24} \\ d_{03} + d_{04} + d_{23} + d_{13} + d_{14} + d_{24} \\ d_{34} + d_{04} + d_{24} + d_{14} \end{bmatrix}}_{\mathbf{d}} \tag{8}$$

In the linear system above, the diagonal entries of A—a_{11}, a_{22}, a_{33} and a_{44}— are the number of paths passing respectively through the edges e_{01}, e_{02}, e_{03} and e_{34}. Also, for example, a_{12} is the number of paths passing through both edges e_{01} and e_{02}, and a_{34} is the number of paths passing through both edges e_{03} and e_{34}. In vector $\mathbf{d}$, in the first entry—$d_{01} + d_{12} + d_{13} + d_{14}$—the indices correspond to the beginning vertex and end vertex of the paths that the edge e_{01} is on, and d_{ij} is the distance between the vertices in the complete weighted graph.

The question is how do we count the number of paths passing through one specific edge or two specific edges in a tree effectively? To answer the first part of the question, let us take one vertex of the tree as the root vertex and consider the tree directed based on that vertex. Let us also denote the descendants of the vertex v_i by D_i. We can say the number of paths passing through edge e_{ij} in a tree is equal to $(|D_j| + 1)(n - (|D_j| + 1))$ where v_j is the bottom vertex—as in further from the root compared to v_i. For example, the number of paths passing through the edge e_{03} in Fig. 2 with v_0 being the root is $(|D_3| + 1)\,(10 - (|D_3| + 1)) = (4 + 1)(10 - (4 + 1)) = 25$.

To answer the second part of the question, that is, to count the number of paths passing through two edges, say, e_{ij} and e_{rs} where $v_j \in D_i$ and $v_s \in D_r$, we do as follows. If $D_j \cap D_s = \emptyset$, the number of paths passing through the two edges is $(|D_j| + 1)(|D_s| + 1)$. For example, the number of paths passing through the edges e_{02} and e_{37} in Fig. 2 is 9 since the descendants of vertices v_2 and v_7 are respectively $D_2 = \{v_4, v_5\}$ and $D_7 = \{v_8, v_9\}$. If for the bottom vertices v_j and v_s, $D_j \cap D_s \neq \emptyset$, then either $D_j \subset D_s$ or $D_s \subset D_j$. For the former, the number of paths passing

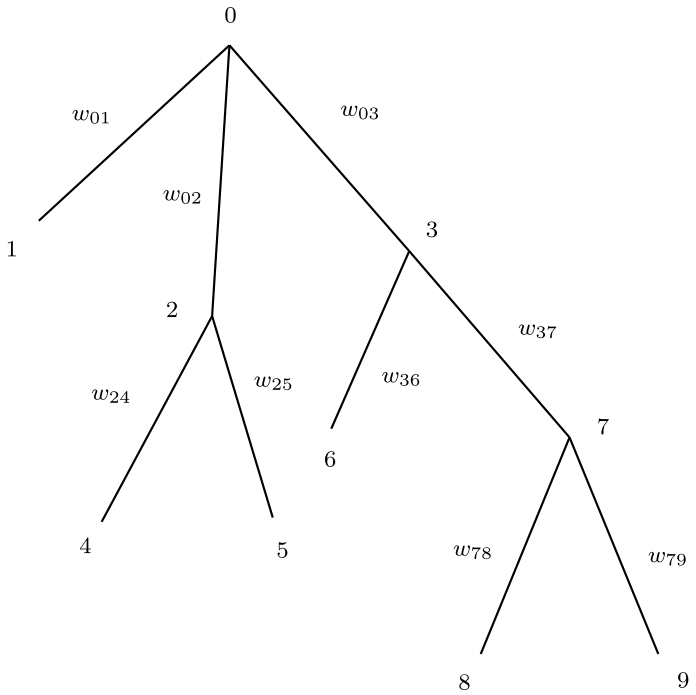


Fig. 2 An example of finding the entries of matrix A on a tree

through both edges is $(|D_j|+1)\,(n-(|D_s|+1))$, and for the latter this value is $(|D_s|+1)\,\big(n-(|D_j|+1)\big)$. For example, the number of paths passing through both edges e_{03} and e_{37} is $(|D_7|+1)(n-(|D_3|+1)) = (2+1)(10-(4+1)) = 15$.

After finding all entries of A, we can find the edge weights using $\mathbf{w} = A^{-1}\mathbf{d}$. Yet, is the matrix A necessarily invertible? In the following, we prove that not only is A invertible, but positive-definite.

Lemma 1 *A is a positive-definite matrix.*

Proof We define the function Z on a spanning tree T as follows. For each edge e_{ij}, we assign a variable v_{ij}. Then we define $Z = \sum_{\substack{m,k \\ m<k}} \left(\sum_{\substack{i,j \\ e_{ij} \in P_{m,k}}} v_{ij} \right)^2$. We can see that the terms $(.)^2$ in Z are the same as those in RSS (Eq. 2). The only difference being the variables w_{ij} are replaced by v_{ij} and the constants d_{ij} are replaced by 0. Z can be written as $Z = \mathbf{v}^\intercal B\mathbf{v} > 0$ where $\mathbf{v}$ is the vector of variables v_{ij} and B is a matrix whose entries are as follows. b_{pq} is the number of terms $(.)^2$ in Z including the variable v_{ij} assigned to the p-th entry of vector $\mathbf{v}$ for $p = q$, and for $p \neq q$, b_{pq} is the number of terms $(.)^2$ including both variables v_{ij} and v_{rs} assigned to the p-th and q-th entries of vector $\mathbf{v}$. Since each term $(.)^2$ denotes a path in T, we can say that b_{pq} is the number of paths including the edge e_{ij} assigned the p-th entry of vector

$\mathbf{v}$ for $p = q$, and for $p \neq q$, b_{pq} is the number of paths including both edges e_{ij} and e_{rs} assigned the p-th and q-th entries of vector $\mathbf{v}$. Thus, $B = A$, and since B is positive-definite, A is also a positive definite matrix. □

Since A is positive-definite, we can use the Cholesky decomposition of A in the form $A = LL^{\intercal}$ where L is a unique lower triangular matrix whose entries are computed by Eqs. 9 and 10. From there, we can solve $Ly = \mathbf{d}$, and then $L^{\intercal}\mathbf{w} = y$ to find the weights. In the following, we discuss how we find the optimal tree.

$$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2} \tag{9}$$

$$L_{ij} = \frac{1}{L_{jj}} \left(A_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk} \right) \tag{10}$$

3 Problem: Tree Structure Optimisation

So far, we discussed how we can find the edge weights for a given tree based on the edge weights in the complete weighted graph. The question is, how can we find the tree with the minimum RSS? We can build n^{n-2} spanning trees on any n number of labelled vertices. That means for as few as 50 labelled vertices, we can have roughly as many spanning trees as the number of atoms in the known universe. Due to the large scale of the problem, we make use of two metaheuristics—in this case, Simulated Annealing (SA) and Iterated Local Search (ILS)—to approximate the optimal tree. Below, we explain SA, and how we use it to approximate the tree.

3.1 Simulated Annealing (SA)

Suppose we have a system with N states—state space of size N. We define an energy function E on the state space. The objective is finding the state with the minimum value of E. For state s, we define the neighbour state s' as one that is obtained by making an alteration in s. Defining s_t as the state of the system at time t, and starting from $t = 0$, we make the transition from $s_t \leftarrow s$ to $s_{t+1} \leftarrow s'$ in two cases:

1. $E(s') < E(s)$.
2. $P(E(s') - E(s), t) > \text{random}(0, 1) \quad \text{if } E(s') > E(s)$.

Otherwise $s_{t+1} \leftarrow s$. In the above, random(0, 1) denotes a number picked uniformly at random in the interval (0, 1). Case 2 points out that we accept a state with a higher energy level than that of its neighbour with a probability that is a function of the value of t and the difference between $E(s)$ and $E(s')$. In other words, the larger the value of $E(s') - E(s)$ or that of t, the less probable the transition from $s_t \leftarrow s$ to $s_{t+1} \leftarrow s'$. It should be noted that case 2 ensures the method does not get stuck in a local minimum—something that would happen if we had only case 1 which is equivalent to a descent-only algorithm. We continue as the above until the system reaches a state that is good enough.

3.2 Iterated Local Search (ILS)

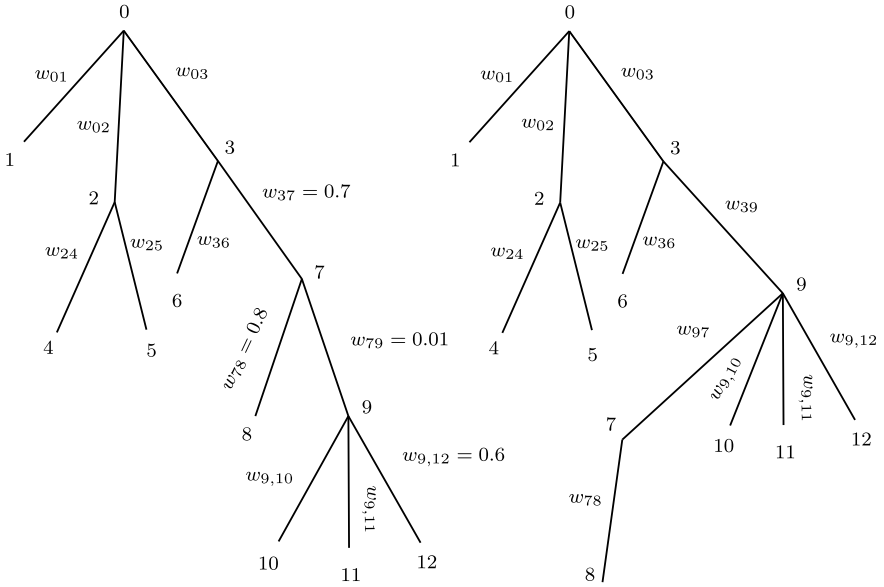
As with SA, we want to minimise the energy function E, but this time, we make the transition from $s_t \leftarrow s$ to $s_{t+1} \leftarrow s'$ only if $E(s') < E(s)$—so far, it is a descent-only algorithm. However, in contrast to a descent-only algorithm, when we get stuck in a local minimum, we restart the algorithm—by modification of the current local minimum—to a new initial system state. Basically, ILS consists of the repeated application of the following two steps:

1. Modification of the current local minimum by kicking it far enough from its current basin
2. Descent to get to a new local minimum

3.3 Tree Structure Change for Optimisation

Before discussing SA and ILS on a tree, let us explain how we make a change in the structure of a given tree in order ot accept or reject the transition between two states—we consider the structure of the tree as the system state. Let T_t be the tree at time t and let us denote its corresponding structure by $T(V, E)$. Let us also denote the structure after change by $T'(V, E')$—the structure that we want to accept or reject. For $v_i \in V$, we denote the neighbours of v_i by $N(v_i)$. We pick one edge $e_{ij} \in E$ (we are going to discuss how we pick this edge in the rest). Then we define set C as $C = N(v_i) \cup N(v_j) \setminus \{v_i, v_j\}$. We pick $v_k \in C$ uniformly at random. If $v_k \in N(i)$, then $E' = E \cup \{e_{jk}\} \setminus \{e_{ik}\}$; otherwise, if $v_k \in N(j)$, then $E' = E \cup \{e_{ik}\} \setminus \{e_{jk}\}$. We denote the former structure change by $SC(T, e_{ij}, e_{jk}, e_{ik})$ and the latter by $SC(T, e_{ij}, e_{ik}, e_{jk})$. In $SC(T, ., ., .)$, the second, third, and forth terms are respectively the picked edge, the edge that is added to, and the edge that is removed from the tree.

How do we pick the edge e_{ij} to make the structure change in the tree? We can either pick it uniformly at random, or we can be biased towards the smaller edges.



(a) Tree before structure change (b) Tree after structure change

Fig. 3 Demonstration of the structure change $SC(T, e_{79}, e_{39}, e_{37})$

We explain the latter in the context of the following example. In the tree in Fig. 3a, suppose that after finding the edge weights, e_{79} has a weight significantly smaller than the rest of the edges. This means that the distance between the vertices in $N(v_7) \setminus v_9$, which are v_3 and v_8, and those in $N(v_9) \setminus v_7$, which are v_{10}, v_{11} and v_{12}, is approximately the sum of the edge weights between them without counting w_{79}. In other words, for $i \in \{v_3, v_8\}$ and $j \in \{v_{10}, v_{11}, v_{12}\}$, $S(P_{i,j}) \approx \sum_{e_{rs} \in P_{ij} \setminus e_{79}} w_{rs}$. Thus, for the next tree structure (the next state), we take one of the vertices—picked uniformly at random—in $N(v_7) \setminus v_9$ or $N(v_9) \setminus v_7$ and connect it respectively to v_9 or v_7. That is to say, in the former, the structure change is $SC(T, e_{79}, e_{9k}, e_{7k})$ where $k \in \{3, 8\}$, and in the letter, it is $SC(T, e_{79}, e_{7k}, e_{9k})$ where $k \in \{10, 11, 12\}$. For example, the resulting tree of $SC(T, e_{79}, e_{93}, e_{73})$ will look like Fig. 3b.

If we prefer edges with smaller weights—whose effect we are going to investigate in the rest— to make the structure change, we want the smaller an edge weight, the more probable to be selected. To this end, for $T(V, E, w)$ over n vertices, we sort the edges according to their weights. Let $\mathbf{e}^s$ denote the column vector of the edges E in the order of their weights. We take a number r uniformly at random from the interval $(0, 1)$. Then the edge that we pick for the structure change is the entry $\lceil r^l n \rceil$ which is $\mathbf{e}^s$ $(e^s_{\lceil r^l n \rceil})$ where $l \in \{z : z \in \mathbb{Z}^+, z > 1\}$.

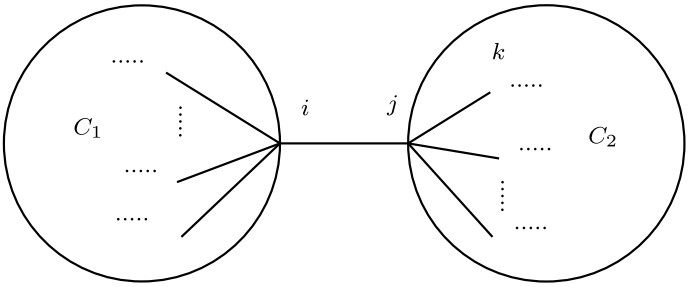


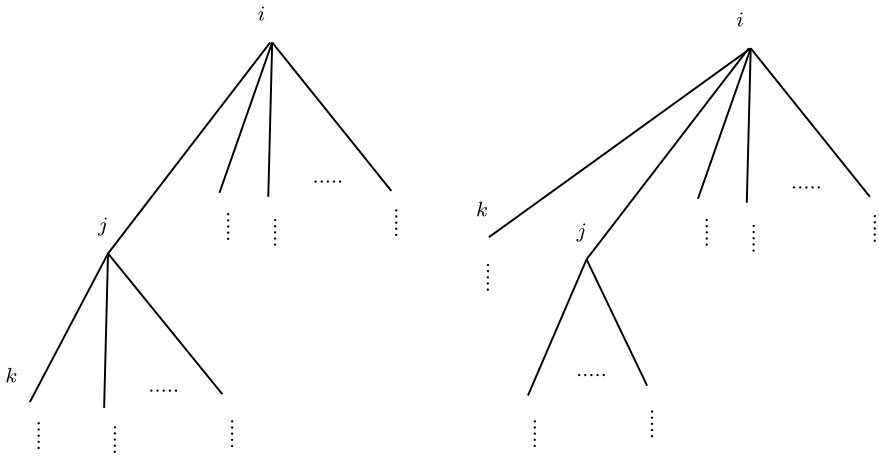
Fig. 4 Tree $T(V, E)$ before the structure change with picked edge e_{ij} connecting components C_1 and C_2, and randomly picked vertex $v_k \in C$

The other thing we investigate before discussing the SA and ILS algorithms on a tree is the change in matrix A and vector $\mathbf{d}$ following the structure change in $T(V, E)$. Should we recompute every entry of A and $\mathbf{d}$ after every structure change? Let us define A' and $\mathbf{d}'$ as the matrix and vector corresponding to $T'(V, E')$.

Lemma 2 *Suppose we have the structure change $SC(T, e_{ij}, ., .)$ resulting in tree $T'(V, E')$. All the entries of A and A' are the same except the rows and columns corresponding to e_{ij}. So are all the entries in $\mathbf{d}$ and $\mathbf{d}'$ except the entry corresponding to e_{ij}. Thus, we only need to recompute the entry in $\mathbf{d}$ and the rows and columns in A corresponding to e_{ij} to obtain A' and $\mathbf{d}'$.*

Proof Consider the tree $T(V, E)$ in Fig. 4 on which we want to make the structure change based on the picked edge e_{ij} and $v_k \in C$ (C as defined in Sect. 3.3). α and β are as defined in Eq. 4 for $T(V, E)$, and the equivalents of them are α' and β' for $T'(V, E')$. If $v_k \in N(v_j) \setminus \{v_i\}$, the structure change is $SC\left(T, e_{ij}, e_{ik}, e_{jk}\right)$.

Let us look at $T(V, E)$ as a directed tree with the root vertex v_i. This tree before and after the structure change is illustrated in Fig. 5. Consider the subgraph $S = G(V, E'')$ in $T'(V, E')$ where $E'' = E' \setminus \{e_{ij}, e_{ik}\}$. It can be seen that every vertex but v_j in this subgraph has the exact same descendants in T' as they have in T. Thus, since $E'' \subset E'$ and $E'' \subset E$, we can say that the number of paths that pass through any edge or any two edges in E'' is the same in T and T'. Similarly, regarding $e_{ik} \in E'$ and $e_{jk} \in E$, $\alpha'_{ik} = \alpha_{jk}$ and $\beta'_{ikrs} = \beta_{jkrs}$ for all $e_{rs} \in E''$. Hence, we see that e_{ij} is the only edge for which $\alpha'_{ij} \neq \alpha_{ij}$ and $\beta'_{ijrs} \neq \beta_{ijrs}$ where $e_{rs} \in E \cap E' \setminus e_{ij}$. $\square$



(a) Tree before structure change T (b) Tree after structure change T'

Fig. 5 Demonstration of the structure change $SC(T, e_{ij}, e_{ik}, e_{jk})$. Only v_j has a different number of descendants in T' than it has in T

As an example, let us see how A changes when we make the structure change $SC(T, e_{79}, e_{93}, e_{73})$ in $T(V, E)$ in Fig. 3a to get $T'(V, E')$ in Fig. 3b. In Eq. 11, we see the rows and columns corresponding to the picked edge e_{79} change from A to A'. On the other hand, we see that the rows and columns corresponding to the removed edge e_{37} in A remain the same in A' (other than entries that belong to the rows and columns corresponding to e_{79} as well) where those rows and columns correspond to e_{39}.

$$
\underbrace{\begin{array}{c|ccccccccccc}
 & e_{01} & e_{02} & e_{03} & e_{24} & e_{25} & e_{36} & e_{37} & e_{78} & e_{79} & e_{9,10} & e_{9,11} & e_{9,12} \\
\hline
e_{01} & 12 & 3 & 8 & 1 & 1 & 1 & 6 & 1 & 4 & 1 & 1 & 1 \\
e_{02} & 3 & 30 & 24 & 10 & 10 & 3 & 18 & 3 & 12 & 3 & 3 & 3 \\
e_{03} & 8 & 24 & 40 & 8 & 8 & 5 & 30 & 5 & 20 & 5 & 5 & 5 \\
e_{24} & 1 & 10 & 8 & 12 & 1 & 1 & 6 & 1 & 4 & 1 & 1 & 1 \\
e_{25} & 1 & 10 & 8 & 1 & 12 & 1 & 6 & 1 & 4 & 1 & 1 & 1 \\
e_{36} & 1 & 3 & 5 & 1 & 1 & 12 & 6 & 1 & 4 & 1 & 1 & 1 \\
e_{37} & 6 & 18 & 30 & 6 & 6 & 6 & 42 & 7 & 28 & 7 & 7 & 7 \\
e_{78} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 12 & 4 & 1 & 1 & 1 \\
e_{79} & 4 & 12 & 20 & 4 & 4 & 4 & 28 & 4 & 36 & 9 & 9 & 9 \\
e_{9,10} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 9 & 12 & 1 & 1 \\
e_{9,11} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 9 & 1 & 12 & 1 \\
e_{9,12} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 9 & 1 & 1 & 12
\end{array}}_{A}
\rightarrow
\underbrace{\begin{array}{c|ccccccccccc}
 & e_{01} & e_{02} & e_{03} & e_{24} & e_{25} & e_{36} & e_{39} & e_{78} & e_{79} & e_{9,10} & e_{9,11} & e_{9,12} \\
\hline
e_{01} & 12 & 3 & 8 & 1 & 1 & 1 & 6 & 1 & 2 & 1 & 1 & 1 \\
e_{02} & 3 & 30 & 24 & 10 & 10 & 3 & 18 & 3 & 6 & 3 & 3 & 3 \\
e_{03} & 8 & 24 & 40 & 8 & 8 & 5 & 30 & 5 & 10 & 5 & 5 & 5 \\
e_{24} & 1 & 10 & 8 & 12 & 1 & 1 & 6 & 1 & 2 & 1 & 1 & 1 \\
e_{25} & 1 & 10 & 8 & 1 & 12 & 1 & 6 & 1 & 2 & 1 & 1 & 1 \\
e_{36} & 1 & 3 & 5 & 1 & 1 & 12 & 6 & 1 & 2 & 1 & 1 & 1 \\
e_{39} & 6 & 18 & 30 & 6 & 6 & 6 & 42 & 7 & 14 & 7 & 7 & 7 \\
e_{78} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 12 & 11 & 1 & 1 & 1 \\
e_{79} & 2 & 6 & 10 & 2 & 2 & 2 & 14 & 11 & 22 & 2 & 2 & 2 \\
e_{9,10} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 2 & 12 & 1 & 1 \\
e_{9,11} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 2 & 1 & 12 & 1 \\
e_{9,12} & 1 & 3 & 5 & 1 & 1 & 1 & 7 & 1 & 2 & 1 & 1 & 12
\end{array}}_{A'}
\tag{11}
$$

Lastly, we want the initial tree—the input tree of the optimisation—to be a random one since we want to make sure that we obtain a reasonable optimal tree regardless of the initial tree. For this purpose, we do as follows.

Algorithm Initial tree

Input:
$MST\ (K_n(V, \mathbf{E}, d))$: MST of the complete weighted graph
Output:
T_0 : the initial tree

$T(V, E) \leftarrow MST(K_n(V, \mathbf{E}, d))$
for $i = 1$ to M **do**
  $e_{ij} \leftarrow$ Picked from E uniformly at random
  $C \leftarrow N(v_i) \cup N(v_j) \setminus \{v_i, v_j\}$
  $v_k \leftarrow$ uniformly picked from C
  if $v_k \in N(i)$ **then**
    $T'(V, E') \leftarrow SC(T, e_{ij}, e_{jk}, e_{ik})$
  else
    $T'(V, E') \leftarrow SC(T, e_{ij}, e_{ik}, e_{jk})$
  end if
  $T \leftarrow T'$
end for
$T_0 \leftarrow T$

In the above, we make a large number M of random structure changes to the minimum spanning tree (MST) of the complete graph. The resulting tree is the initial tree that we implement the optimisation algorithms on.

3.3.1 SA on Tree

Below is the SA algorithm on a tree with the picked edge e_{ij} for making a structure change biased towards smaller edges. The larger the value of l, the more biased towards smaller edges. The edge can also be picked uniformly at random. Also, end_time denotes the amount of time that we want to run the algorithm, and $time$ is the amount of time already passed. The function that specifies probability is $P(RSS', RSS, t)=a_1 e^{-a_2(\ln t)^{a_3} \frac{RSS'-RSS}{RSS'}}$ where the parameters a_1, a_2 and a_3 are tuned according to how often we are willing to accept a transition with a larger objective function value RSS' compared to the current one RSS.

3.3.2 ILS on Tree

In contrast to SA, as shown below, the picked edge e_{ij} in ILS is always picked uniformly at random. The reason being we hope to try every possible structure change to "make sure" the function RSS is stuck at a local minimum. The variable $count$ controls whether we have got stuck at a local minimum as we explain in the following. For any picked edge e_{ij}, the number of structure changes that we can make depends on $|C_i| = |N(v_i) \setminus \{v_j\}|$ and $|C_j| = |N(v_j) \setminus \{v_i\}|$. If we remove the edge e_{ij} from T, the resulting graph $G(V, E \setminus \{e_{ij}\})$ consists of two trees T_i and T_j where

Algorithm 2 SA on tree

Input:
K_n : the complete weighted graph
T_0 : initial tree structure
Output:
$T_{optimal}$: the optimal tree

$t \leftarrow 0$
$T(V, E) \leftarrow T_0$
$\mathbf{w} \leftarrow$ Solve $A\mathbf{w} = \mathbf{d}$ via Cholesky decomposition
$RSS \leftarrow RSS(T, K_n)$
$RSS_{best} \leftarrow RSS(T, K_n)$
while $time < end_time$ **do**
  $t \leftarrow t + 1$
  $\mathbf{e}^s \leftarrow$ column vector of edges in E sorted based on weight
  $e_{ij} \leftarrow e^s_{\lceil r^l n \rceil}$
  $C \leftarrow N(v_i) \cup N(v_j) \setminus \{v_i, v_j\}$
  $v_k \leftarrow$ uniformly picked from C
  if $v_k \in N(i)$ **then**
    $T'(V, E') \leftarrow SC(T, e_{ij}, e_{jk}, e_{ik})$
    $\mathbf{w}' \leftarrow$ Solve $A'\mathbf{w}' = \mathbf{d}'$ via Cholesky decomposition
  else
    $T'(V, E') \leftarrow SC(T, e_{ij}, e_{ik}, e_{jk})$
    $\mathbf{w}' \leftarrow$ Solve $A'\mathbf{w}' = \mathbf{d}'$ via Cholesky decomposition
  end if
  $RSS' \leftarrow RSS(T', K_n)$
  if $RSS' < RSS_{best}$ **then**
    $RSS_{best} \leftarrow RSS'$
    $T_{optimal} \leftarrow T'$
  end if
  if $RSS' < RSS$ **then**
    $A \leftarrow A'$
    $\mathbf{d} \leftarrow \mathbf{d}'$
    $RSS \leftarrow RSS'$
    $T \leftarrow T'$
  else if $P(RSS', RSS, t) > random(0, 1)$ **then**
    $A \leftarrow A'$
    $\mathbf{d} \leftarrow \mathbf{d}'$
    $RSS \leftarrow RSS'$
    $T \leftarrow T'$
  end if
end while

$v_i \in T_i$ and $v_j \in T_j$. We assume the average degree of a tree to be 2; thus, we assume the degree of both v_i and v_j to be 2. With this assumption, the number of possible structure changes based on the edge e_{ij} is two, so for the whole tree, we estimate it at $4n$. If we try structure changes on a tree uniformly at random, the average number of times that we need to try all possible structure changes is $4nH_{4n}$—based on the well-known Coupon collector's problem—where H_{4n} is the $4n$-th harmonic number defined as $H_k = \sum_{i=1}^{k} \frac{1}{i}$. That is why we set $4nH_{4n}$ as the threshold of $count$; after

which, we modify the current local minimum by $RT(T, n)$. Here, $RT(T, M)$ denotes the random tree that is obtained by M random structure changes on tree T.

Algorithm ILS on tree

```
Input:
  K_n : the complete weighted graph
  T_0 : initial tree structure
Output:
  T_optimal : the optimal tree

T(V, E) ← T_0
w ← Solve Aw = d via Cholesky decomposition
RSS ← RSS(T, K_n)
RSS_best ← RSS(T, K_n)
while time < end_time do
  e_ij ← Picked from E uniformly at random
  C ← N(v_i) ∪ N(v_j) \ {v_i, v_j}
  v_k ← uniformly picked from C
  if v_k ∈ N(i) then
    T'(V, E') ← SC(T, e_ij, e_jk, e_ik)
    w' ← Solve A'w' = d' via Cholesky decomposition
  else
    T'(V, E') ← SC(T, e_ij, e_ik, e_jk)
    w' ← Solve A'w' = d' via Cholesky decomposition
  end if
  RSS' ← RSS(T', K_n)
  if RSS' < RSS_best then
    RSS_best ← RSS'
    T_optimal ← T'
  end if
  if RSS' < RSS then
    count ← 0
    A ← A'
    d ← d'
    RSS ← RSS'
    T ← T'
  else
    count ← count + 1
    if count > 4nH_4n then
      T ← RT(T, n)
      count ← 0
      w ← Solve Aw = d via Cholesky decomposition
      RSS ← RSS(T, K_n)
      if RSS < RSS_best then
        RSS_best ← RSS
        T_optimal ← T
      end if
    end if
  end if
end while
```

4 Results

We applied SA and ILS as described in Sects. 3.3.1 and 3.3.2 to evaluate the performance of these two metaheuristics in different scenarios. The distance values in K_n are based on stock-correlation data. We evaluated whether bias towards smaller edges has any advantage in SA over no bias in SA. Then we compared the performance of SA and ILS factoring in the underlying distribution of K_n edge distances in different random samples of stocks.

4.1 Biased Versus Unbiased SA

If we set the value of l to 1 in SA, the algorithm picks an edge uniformly at random at each stage. In other words, the algorithm is not biased towards smaller edges. Otherwise, the larger the value of l, the more biased towards smaller edges the algorithm is. Below, we outline the result of the biased and unbiased runs of SA for random sample sizes of 20, 30, and 50. For each sample, we do the biased SA by setting $l = 6$. Also, we ran the biased and unbiased SA on each sample 10 times; each starting from a random tree as per the initial tree construction outlined in Sect. 3.3.

In Table 1, we ran the algorithm on the three random samples for ten min, two hours, and 18 hours respectively. The values in the table are for the RSS function defined in Eq. 1—according to which we evaluate the performance of the algorithm. We can see that other than the random sample with size 20, in the other samples, the biased SA has an overall better performance. For the random sample with size 20,

Table 1 Biased versus Unbiased SA on different random sample sizes. For each tree, the metaheuristic with a better performance has been highlighted

	Sample size					
	20		30		50	
Run	Unbiased	Biased	Unbiased	Biased	Unbiased	Biased
1	4.26687812	4.26687812	12.7055242	12.7055242	32.3461206	**32.1886367**
2	4.26687812	4.26687812	12.7798981	**12.7055242**	32.4564873	**31.7803389**
3	**4.26687812**	5.0111434	**12.7207733**	12.7464911	32.60437806	**31.6861588**
4	4.26687812	4.26687812	12.7055242	12.7055242	**31.85676394**	32.0511834
5	4.26687812	4.26687812	12.7055242	12.7055242	32.39138879	**31.6578321**
6	4.26687812	4.26687812	12.7343806	**12.7055242**	32.28023281	**32.0794777**
7	4.26687812	4.26687812	**12.7361441**	12.7615612	32.08112489	**31.9106985**
8	**4.26687812**	4.37441678	12.7615612	**12.7055242**	**31.92894628**	32.0031474
9	4.27373433	**4.26687812**	12.7207733	**12.7055242**	32.36184357	**31.9668845**
10	4.26687812	4.26687812	**12.7055242**	12.7207733	32.26931596	**31.9372013**
Average	**4.26756374**	4.35205851	12.7275627	**12.7167495**	32.25766022	**31.9261559**

Table 2 SA versus ILS on a complete random weighted graph with low dispersion of distances. For each tree, the metaheuristic with a better performance has been highlighted

	Tree size					
	20		30		50	
Run	SA	ILS	SA	ILS	SA	ILS
1	5.291951597	**4.84414153**	9.501749529	**8.4082242**	24.8579367	**16.8823573**
2	8.21566953	**4.84414153**	11.66835384	**8.4082242**	26.7081655	**16.8823573**
3	7.797470793	**4.84414153**	11.16650355	**7.97443788**	19.6536142	**16.8823573**
4	6.875995126	**4.84414153**	11.66835384	**7.97443788**	25.5941918	**16.8823573**
5	6.875995126	**4.84414153**	8.408224203	**7.97443788**	26.284257	**16.8823573**
6	6.019906558	**4.84414153**	12.98953771	**7.97443788**	30.2829294	**16.8823573**
7	7.016393465	**4.84414153**	10.3933439	**7.97443788**	34.8261135	**16.8823573**
8	7.016393465	**4.84414153**	11.80185307	**7.97443788**	31.0322914	**16.8823573**
9	**4.844141529**	5.2919516	13.02265027	**7.97443788**	31.1982311	**16.8823573**
10	7.016393465	**4.84414153**	10.58531443	**7.97443788**	26.265805	**16.8823573**
Average	6.697031065	**4.88892254**	11.12058844	**8.06119515**	27.6703536	**16.8823573**

possibly since there are only 19 edges, the effect of biased or unbiased edge selection is less. Also, judging by how many times all the runs yield the same RSS value—if that value is the minimum compared to the other runs—there is not a considerable difference between biased and unbiased SA for sample size of 20. The reason being, there are nine minimum values for unbiased and eight for biased. Whereas, for sample size of 30, four of the runs in unbiased SA and seven in biased SA yield the minimum value of RSS. When the sample size is 50, the performance gap between the unbiased and biased SA becomes wider—apparently because the importance of biased edge selection increases as the number of edges in the network increase. It should be noted that we ran the unbiased and biased SA on other samples, and we got similar results. Thus, apparently, biased edge selection has a slight advantage over unbiased. For the remainder, wherever we mention SA, we mean the biased SA.

4.2 SA Versus ILS

As with the comparison between unbiased and biased SA, we compared the performance of SA and ILS based on ten runs of them over the random same samples. Look at Tables 2 and 3 for a performance comparison of SA and ILS. In Table 2, again, we ran the algorithms on random samples of size 20, 30 and 50. It can be seen that the performance of ILS is much better than that of SA. However, in Table 3, we can see that there is no apparent difference between SA and ILS performance.

The reason for performance inconsistency of SA in Tables 2 and 3 seems to be the distance dispersion of samples (edge weights of K_n) used in each of them. For example, for the random sample of size 50 in each table, the dispersion of distances

Table 3 SA versus ILS on a complete weighted graph with high dispersion of distances. For each tree, the metaheuristic with a better performance has been highlighted

	Tree size					
	20		30		50	
Run	SA	ILS	SA	ILS	SA	ILS
1	5.75665216	5.75665216	**12.7055242**	12.8162316	32.18863674	**31.8890846**
2	5.75665216	5.75665216	**12.7055242**	12.7495148	**31.78033885**	31.8140415
3	5.75665216	5.75665216	**12.7464911**	12.8550396	**31.68615883**	31.8986455
4	5.75665216	5.75665216	**12.7055242**	12.8345686	32.0511834	**31.9636283**
5	5.75665216	5.75665216	**12.7055242**	13.0379292	**31.65783212**	31.6641679
6	5.75665216	5.75665216	**12.7055242**	12.832506	32.07947769	**31.7900896**
7	5.75665216	5.75665216	**12.7615612**	12.8750193	**31.91069847**	32.0325964
8	5.75665216	5.75665216	**12.7055242**	12.8785592	**32.00314739**	32.2380063
9	5.75665216	5.75665216	**12.7055242**	12.7207733	**31.96688445**	31.9796543
10	5.93707406	**5.75665216**	**12.7207733**	12.8472376	31.93720134	**31.792651**
Average	5.77469435	**5.75665216**	**12.7167495**	12.8447379	31.92615593	**31.9062565**

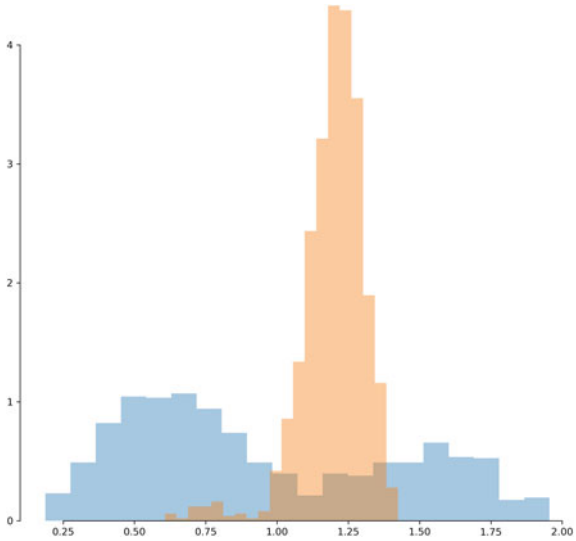


Fig. 6 Dispersion of sample of size 50 in Tables 2 and 3

is illustrated in Fig. 6. It can be seen that for random samples with high dispersion, SA and ILS have a similar performance while for those with low dispersion, ILS has a much better performance than SA. It is noteworthy that for distance values with low dispersion, both biased and unbiased SA have a bad performance. The reason possibly being, when distance values are close to each other, smaller distance values are not considerably smaller than the large distance values.

5 Conclusion

We have presented a scheme to optimise the edges weights and structure of a tree to approximate a complete weighted graph using a measure involving the path distances in the tree. We have proposed a very efficient way of computing modifications to the tree that assist with local search metaheuristics, and evaluated the performance of two of these: SA and ILS. For SA, we demostrated that in each step of the metaheuristic, being biased towards the smaller edge weights when picking a random edge might have a slight advantage over unbiased picking of edges. The results also show that when the dispersion of the edge weights in the complete weighted graph is high, there is no apparent difference between the performance of SA and ILS. However, when is dispersion is low, the performance of ILS is much better than that of SA.

References

1. Birch, J., Pantelous, A.A., Soramäki, K.: Analysis of correlation based networks representing dax 30 stock price returns. Comput. Econ. **47**(4), 501–525 (2016)
2. Boginski, V., Butenko, S., Pardalos, P.M.: Statistical analysis of financial networks. Comput. Stat. Data Anal. **48**(2), 431–443 (2005)
3. Desper, R., Gascuel, O.: Theoretical foundation of the balanced minimum evolution method of phylogenetic inference and its relationship to weighted least-squares tree fitting. Mol. Biol. Evol. **21**(3), 587–598 (2004)
4. Felsenstein, J., Felenstein, J.: Inferring Phylogenies, vol. 2. Sinauer Associates Sunderland, MA (2004)
5. Han, D., et al.: Network analysis of the Chinese stock market during the turbulence of 2015–2016 using log-returns, volumes and mutual information. Physica A: Stat. Mech. Appl. **523**, 1091–1109 (2019)
6. Kocheturov, A., Batsyn, M., Pardalos, P.M.: Dynamics of cluster structures in a financial market network. Physica A: Stat. Mech. Appl. **413**, 523–533 (2014)
7. Mantegna, R.N.: Hierarchical structure in financial markets. European Phys. J. B-Condens. Matter Complex Syst. **11**(1), 193–197 (1999)
8. Narasimhan, G., Smid, M.: Geometric Spanner Networks. Cambridge University Press (2007)
9. Onnela, J.P., Chakraborti, A., Kaski, K., Kertesz, J., Kanto, A.: Asset trees and asset graphs in financial markets. Physica Scripta **2003**(T106), 48 (2003)
10. Tumminello, M., Aste, T., Di Matteo, T., Mantegna, R.N.: A tool for filtering information in complex systems. Proc. Nat. Acad. Sci. **102**(30), 10421–10426 (2005)
11. Tumminello, M., Coronnello, C., Lillo, F., Micciche, S., Mantegna, R.N.: Spanning trees and bootstrap reliability estimation in correlation-based networks. Int. J. Bifurc. Chaos **17**(07), 2319–2329 (2007)

Simulation of Diffusion Processes in Bimetallic Nanofilms

Vladimir Myasnichenko, Rossen Mikhov, Leoneed Kirilov, Nickolay Sdobnykov, Denis Sokolov, and Stefka Fidanova

Abstract Surface diffusion plays a crucial role in the formation of the shape and morphology of growing nanoparticles and nanofilms. Bulk heterodiffusion occurs at uneven (irregular) concentrations of several metals, in the presence of free energy in the system. Atoms of each sort tend to be evenly distributed in volume and form mixed bonds. In this paper, we propose an approach for modeling diffusion processes in nanoalloys by the vacancy mechanism. It is a hybrid Monte Carlo approach based on computing the probability for a transition of each atom belonging to the first three coordination spheres. The energy of the system is computed with a tight binding potential. The efficiency of the approach is demonstrated simulating Au–Ag bimetallic nanofilms with a different number of vacancies in the crystal lattice and a different temperature.

1 Introduction

As it is known, atom ordering in the process of forming nanoalloys is one of the factors defining the structure and properties of the materials. In particular, in the case of bimetallic nanostructures, atomic diffusion processes can strongly affect their properties, because they are often also determined precisely by the mutual arrangement of the alloy components. Knowing the mechanism of this process helps to obtain materials with preset properties. At present this kind of problem is extensively studied—[15]. An example is welding due to the mutual diffusion at the atomic level of the surfaces of parts to be welded (diffusion welding)—[13]. Surface diffusion

V. Myasnichenko · N. Sdobnykov · D. Sokolov
Tver State University, Tver, Russia
e-mail: viplabs@yandex.ru

R. Mikhov · L. Kirilov · S. Fidanova (✉)
Institute of Information and Communication Technologies—Bulg, Academy of Sciences, Sofia, Bulgaria

L. Kirilov
e-mail: 1_kirilov_8@abv.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_11

plays a crucial role in the formation of the shape and morphology of growing nanoparticles and nanofilms. Bulk heterodiffusion occurs at uneven (irregular) concentrations of several metals, in the presence of free energy in the system. Atoms of each sort tend to be evenly distributed in volume and form mixed bonds.

The goal of this work is to simulate the effect of the temperature and the relative number of vacancies on the diffusion/segregation processes in an Au–Ag nanofilm using a Monte Carlo method. Atoms of gold and silver alternate along the Y-axis, forming distinct metal interfaces. To achieve the goal we propose an approach for modeling diffusion processes in nanoalloys by the vacancy mechanism. The approachis based on computing the probability for a transition of each atom belonging to the first three coordination spheres. The potential energy of the system is computed by the tight-binding potential of Cleri–Rosato (1993). At present, the many-body interatomic interaction potentials for metals and their binary compounds are well developed, which allows us to simulate the processes of diffusion and phase transitions in bimetallic nanosystems with a high degree of accuracy—[12, 14]. In addition, we have experience modeling using the molecular dynamics method with the tight binding potential. Surface diffusion and structural relaxation processes were studied in mono-metallic gold and copper nanoparticles up to 10,179 atoms in size—[2] and bimetallic Cu–Au particles—Myasnichenko and Starostenkov [19] and Au–Ag particles—[21]. In particular, when modeling ultrafast melt cooling for clusters of different size, an analysis was made of the nucleation of the crystalline phase in nanoclusters.

2 Literature Overview

Nanostructure modeling is aimportant subject from a theoretical and practical point of view. A lot of scientific groups work on developing intelligent algorithms that can simulate the nanostructure of different materials. Mao et al. [18] reported in their work a high viscosity induced nano-diffusive layer for locally printing nanopolymer wires onto a substrate. Using a millimeter-sized fluidic device, a reduced diffusive layer on the nanoscale was achieved by increasing the viscosities of injected solutions.

Zakharov et al. [27] study structural changes in the boundary of some bi-metals when passing shock after-cascade waves. Such structural changes can significantly affect the physical and chemical properties of bimetallic particles. Abraham et al. [1] propose four bimetal Al–Ni nano-powders where compositions varied from 5 to 45% of Nickel are synthesized by the explosion of electrically heated twisted pure Al and Ni wires in Argon. The nano-powders are characterized using electron microscopy, x-ray diffraction, and thermal analysis. Materials are ignited using an electrically heated filament coated with powder and electrostatic discharge (ESD).

In Makarov et al. [17] the formation of a diffusion zone at the Ni/Al interface is studied. The patterns of the formation of the diffusion zone are studied

on samples with different time intervals of annealing. Ren et al. [22] study nano-diffusion bonding of Ti2AlNb to Ni-basedsuperalloy. Raising bonding temperature could enhance metallurgical bonding and improve bonding quality.

Babicheva et al. [3] investigate the effect of grain boundary segregations on martensitic transformation temperatures in NiTi bi-crystals. The segregation is simulated by replacing the monoatomic Ti or Ni layer in the grain boundary with a Ni or Ti layer, respectively. Bashkova et al. [4] perform simulation of melting/crystallization processes of Ag nanoparticles with the help of molecular dynamics. The influence of various conditions of crystallization on the formation of the internal structure in Ag nanoclustersis also investigated. Galashev et al. [7] study molecular dynamics of the stability of aluminium coatings on iron. To describe the diffusion in a crystal using the molecular dynamics model, a temperature-dependent correction to the activation energy is calculated. The problem of the solidification of pure nanoparticles, Ag, Au, and Ag–x%Aunanoalloysare studied at different cooling rates by Davoodi et al. [6].

Yuldasheva et al. [26], using Monte Carlo simulation, explored the phase transitions concerning magnetoelectric interaction at the interface in the superlattices of multiferroic. Zolnikov et al. [30] studied atomic mechanisms of high-speed migration of symmetric tilt grain boundaries in nanocrystalline Ni. They used the periodic boundary conditions to prevent the rotations of the grains. Zorya et al. [31] studied the interaction of impurity atoms of light elements with self-interstitials in fcc metals by the molecular dynamics method. It is shown that the self-interstitial atom in fcc metals migrates through at least two mechanisms: by the displacement and rotation and by the crowdion mechanism.

In the study of structural transformations in nanoparticles, their specific features as compared with the corresponding bulk phases are of interest from both scientific and applied points of view, including the possibility of creating new metals with predefined specific properties. The essential step in studyingmetal clusters is to locate their stable geometrical structures. In the case of metallic nanoparticles, the reliability of the simulation results is primarily determined by the interatomic interaction potential used. The analytical form of potential energy is created in such a way that it contains a specific set of independent parameters. Using these parameters, the potentials are fitted to the experimental data or ab initio data of the calculations. When describing metal nanostructures, it is legitimate to use the many-body form of the potential energy function, which takes into account the banded nature of the metal bonds.

The main advantage of the many-body interaction potential, as compared with the simple pair potential, is its ability to reproduce some of the main features of metallic systems better. A relatively simple way to describe the atomic and electronic structure is tight-binding (TB)—[5, 10], in which the ion-ion interaction is described taking into account the banded nature of the bond and is supplemented by a short-range pair repulsion potential. The method is based on the fact that a significant group of properties of transition metals can be completely determined based on the density of states of external d-electrons. Despite the simple, functional form, the strong-binding model describes well enough the elastic properties, defect characteristics, and melting for a wide range of fcc and hcp metals. In our opinion, this scheme is

one of the most suitable for computer analysis of small metal particles consisting of several hundreds or thousands of atoms. There exist different approaches: the Gupta potential function has been widely applied to the study of face-centered-cubic structures—[5], hexagonal closed packed—[15], transition metals and their alloy clusters—[9]. The stable geometrical structure of six metallic clusters is optimized by a dynamic lattice search method—[25].

In the contact region of two dissimilar metals, a process of mutual diffusion occurs, the result of which is the formation of a transition region, the so-called diffusion zone—[15]. Obviously, the diffusion zone is characterized by a concentration gradient and a gradient of chemical potential due to the inhomogeneous distribution of system components. As a result of this, there is a diffusion movement of atoms, which ultimately leads to the establishment of a phase composition in the entire volume of the sample, which is determined by the state diagram. However, at the nanoscale, appropriate size effects should be taken into account, in particular, at the interface between components—[23].

Moreover, mutual diffusion in binary nanocrystalline systems consisting of two chemical elements is most often provided by a vacancy mechanism—[28]. In this review, an analysis of the causes, driving forces, and mechanisms of pore formation and disappearance in nanoparticles with a closed geometry is described in detail. Taking into account the analyzed effects in the framework of the quasi-stationary approximation, several phenomenological models have been proposed for describing the evolution of nanoshells of solid solutions and intermetallic compounds. A three-dimensional computer simulation model based on the Monte Carlo method is created, which allows studying at the atomic level the competition of these effects, in particular, segregation and its effect on the stability of hollow nanoshells, as well as the effect on the nature of pore formation of temperature and particle size. The topics considered are useful in studying diffusion problems and phase transformations in general. In addition, there are a number of works in which the Monte Carlo method is used to model technological processes that are based on the diffusion process, for example, the diffusion welding process—[16, 24].

To date, there are also some experimental studies on the properties of the interface of nanoscale samples, including under various external conditions and methods for producing the samples under study. In particular, the work of Jahangiri [11] can be noted. This study investigates the effect of aging method and the presence or absenceof external stress on the aging behavior of Cu/Al diffusion couples. Hodak et al. [8] demonstrate the synthesis of bimetallic AucoreAgshell and AgcoreAushell-nanoparticles using radiation chemistry. It is noted that interdiffusion of Ag and Au inthe core–shell nanoparticles also occurs at temperatures that are much lower than the expected melting point. Zhang et al. [29] used SEM, EDS and XRD characterizations to analyze improvement of bonding strength, diffusion behavior between Au and Ag, and structure of the Au layer.

However, when choosing a research method, we note that to date, the further development and application of computer modeling methods is of particular relevance. When used correctly for the nanometer range, computer simulation methods may well compete with direct experiment methods. Computational experiments, as

a rule, are rather laborious, but to a much lesser extent than similar direct experiments. And they are much less expensive than direct experiments. This makes their application relevant both as a stage of research preceding direct experiments, and for an independent examination of the results of direct experiments. Thus, the study of the diffusion process in an Au–Ag nanofilm using the Monte Carlo method is an important task, the results of which help to understand the processes and patterns occurring in the diffusion contact zone of two nanosized metal samples.

3 Proposed Approach

The framework for the Monte Carlo simulation isbased on Korznikova et al. [13], Lobzenko et al. [16], with energy calculation and other details of the algorithm based on our previous work in Myasnichenko et al. [20, 21]. The lattice is given by a block of an fcc crystal. Each node of the lattice contains an atom, with the exception of a few vacancies, required for modeling diffusion.

Energy of the system is expressed by a multi-particle TB potential by Cleri–Rosato (1993) of the following form:

$$E = \sum_i \left(\sum_{j \neq i} E_{ij,ab} - \sqrt{\sum_{j \neq i} B_{ij,ab}} \right) \tag{1}$$

$$E_{ij,ab} = A_{ab} \exp\left(-p_{ab} \left(\frac{r_{ij}}{r_{0,ab}} - 1 \right) \right) \tag{2}$$

$$B_{ij,ab} = \xi_{ab}^2 \exp\left(-2q_{ab} \left(\frac{r_{ij}}{r_{0,ab}} - 1 \right) \right), \tag{3}$$

where i ranges over all atom positions; j ranges over all atom positions other than i but within distance R_{cut} from i; a and b represent the species of the atoms at i and j; $E_{ij,ab}$ and $B_{ij,ab}$ are the repulsive and binding components, respectively, of the potential due to the atoms at i and j; r_{ij} is the distance between the atoms; $r_{0,ab}$, A_{ab}, p_{ab}, ξ_{ab}, q_{ab} are constants particular to the chemical elements under consideration. R_{cut} is the maximum distance beyond which the interaction is assumed to be zero. We use a value for R_{cut} corresponding to five coordination spheres.

Monte Carlo simulation proceeds by iterating a cycle. At each iteration, one vacancy is chosen at random and filled by the jump of a neighboring atom. *Neighboring* here is defined as being within a radius of three coordination spheres. The neighboring atom is also chosen at random, but with a relative probability given by:

$$P = \exp\left(-\frac{\Delta E}{kT} \right), \tag{4}$$

where ΔE is the energy difference for the atom jumping into the vacancy, T is the temperature of the system, and k is the Boltzmann constant. P is calculated separately for each candidate neighbor, and a random number in the interval between zero and the sum of all P s determines which atom will be chosen.

Neighbor lists and R_{cut}-vicinities are pre-computed, for both atoms and empty nodes, along with values for $E_{ij,ab}$ and $B_{ij,ab}$ according to (2) and (3), for all possible combinations of i, j, a and b. Values under the square root of (1) are also memoized and kept updated throughout the algorithm. This results in very fast evaluation of the energy function, allowing hundreds of thousands of energy calculations per second per CPU core.

4 Experiments

Our test case is an XY-periodic lattice with 12,000 nodes corresponding to the fcc Au lattice, filled with Au and Ag atoms in proportion 1:1. There is a small dimensional mismatch between the two metals. The form of the initial configuration is a block having the (100) surface perpendicular to the Z axis, with the periodic boundary conditions along the X and Y axes. Two kinds of initial conditions (long-range orders) are tested. In «random» mode, each trial starts from a different fully random configuration (e.g. Fig. 1a). In «sandwich» mode, all trials start from a fixed configuration

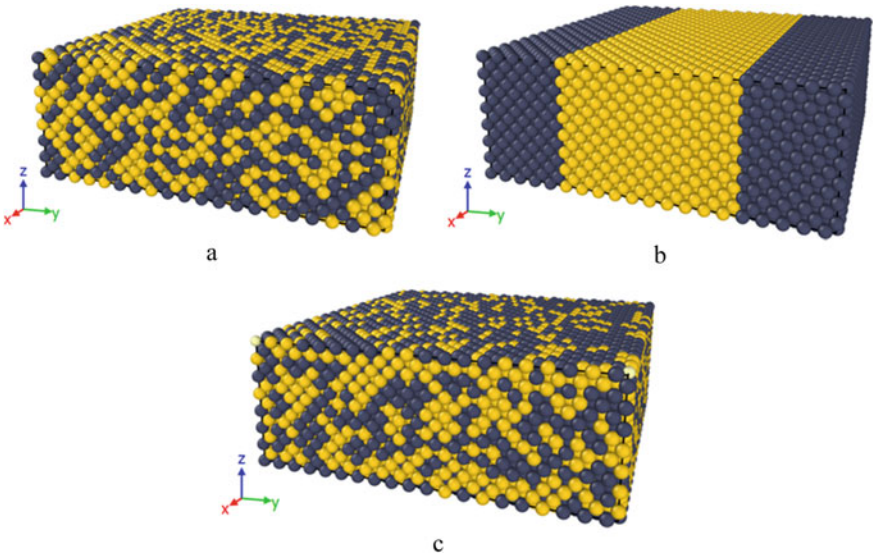


Fig. 1 Schematic view of the investigated Au–Ag nanofilm (Au atoms are yellow): **a** «random» configuration at the temperature of 0 K; **b** «sandwich» configuration at the temperature of 0 K; **c** final configurationat 1200 K

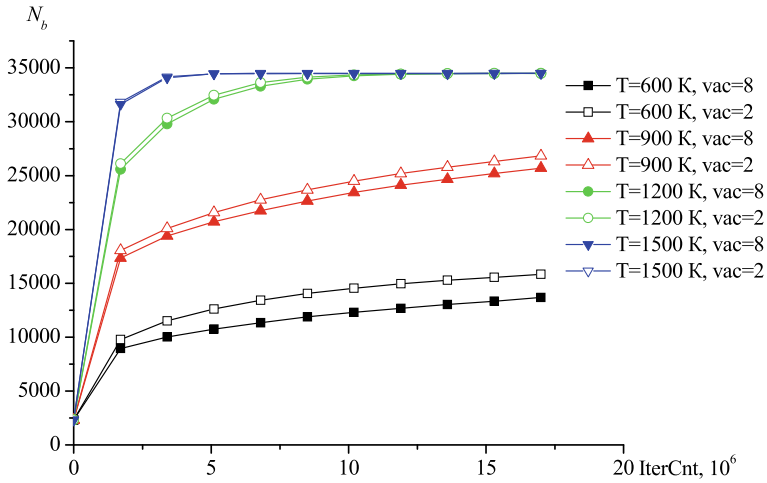


Fig. 2 Dependence of the number of Au–Ag bonds (N_b) on the number of iterations (*IterCnt*) at various temperatures and vacancy concentrations for the «sandwich» configuration

representing plots of gold and silver alternating along the Y axis, forming distinct metal interfaces (Fig. 1b).

To simulate diffusion, we inject from 2 to 8 vacancies by deleting an equal number of Ag and Au atoms, which we choose at random each time. We test several temperatures: 600, 750, 900, 1050, 1200 K with «random» configurations, and 600, 900, 1200, 1400, 1600 K with the «sandwich»configuration. Since the results for 1400 K and 1600 K were almost identical, to prevent clutter in the figures we show their average as one curve under the label T = 1500 K. Each trial is launched for 17 million iterations of the Monte Carlo cycle. Since the algorithm is non-deterministic, we repeated each trial 30 times and present all results in terms of their averages.

Figure 2 is a plot of the number of mixed Au–Ag bonds N_b with the number of iterations *IterCnt* (total number of vacancy jumps) for the «sandwich» configuration. The analysis shows that with increasing temperature, the number of bonds increases nonlinearly. For any given temperature, the condition $N_{b(vac=2)} > N_{b(vac=8)}$ is always satisfied, and with increasing temperature this numerical difference decreases to practically level out at $T = 1500$K. The behavior of the specific potential energy at different *IterCnt*varies in a more sophisticated way (Fig. 3). In particular, $\Delta U_{(T=1200K)} < \Delta U_{(T=1500K)}$ already starting from $8.5{\cdot}10^6$ iterations, i.e. the most energetically beneficial internal structure is formed. As for the number of bonds, the dependence of the specific potential energy more weakly registered on the number of vacancies with increasing temperature.

Figures 4, 5 show the corresponding distributions of Ag atoms in the Y and Z directions (see Fig. 1) for a different number of vacancies. It was found that the resulting profiles are sufficiently symmetrical. With increasing temperature, the profile in the Y direction almost straightens at the equiatom level of 50% for both «random» and «sandwich» configurations for all studied vacancy concentrations.

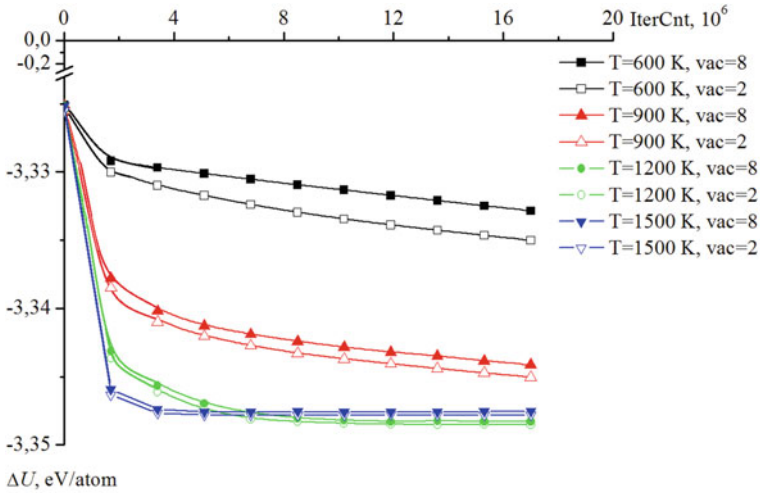


Fig. 3 Dependence of the specific potential energy ΔU on the number of iterations (*IterCnt*) at various temperatures and vacancy concentrations for the «sandwich» configuration

The profile in the Z direction at any temperature has two minimums symmetric in position and magnitude (Au-rich near-surface layers) and two groups of maxima: directly surface layers and layers lying at a depth of 3–4 atomic rows. Moreover, the profile view in the Z direction is also identical in structure between the «sandwich» and the «random» configurations (see Fig. 6).

The main difference is the size of the zone where the fraction of Ag atoms is practically unchanged. Even this difference disappears at $T = 1200$ K. At $T = 600$ K, the all η_{Ag} values stay close to the starting configuration (see Fig. 6).

5 Conclusion

In this paper, we present amethod for simulating diffusion processes by the vacancy mechanism in bimetallic nanofilms. Weexperiment with 2–8 vacancies, working with different temperatures and a fixed number of Monte Carlo iterations. The process is controlled by the probability for a transition of each atom belonging to the first three coordination spheres around the vacant node. The potential energy of the system is computed by a semiempirical tight binding potential. The tests are performed on an XY-periodic lattice corresponding to the 2D fcc crystal, filled with Au and Ag atoms in proportion1:1. The following results can be outlined:

(1) The temperature during the diffusion process is a more critical factor than the initial number of vacancies; with increased temperature, the effect of the difference in the number of vacancies for the number of mixed Au–Ag bonds is almost leveled.

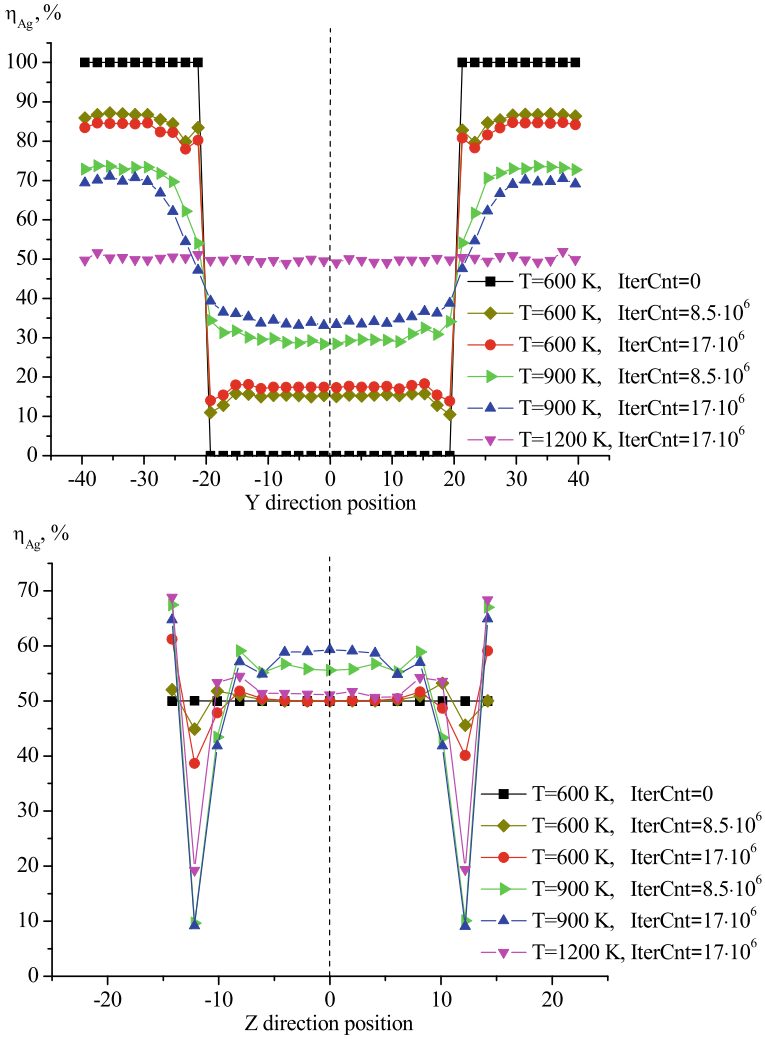


Fig. 4 Distribution of Ag atoms along selected directions for the «sandwich» configuration with the vacancy concentration 2/12000

(2) Due to the statistically identical distribution of atoms and mixed bonds, the short-range order in the resulting bimetallic alloys comes also to be the same. Therefore, the final values of the specific internal energy of a 2D nanoalloy weakly depend on the number of vacancies.

(3) In all final configurations, vacancies move to the nanofilm surface during diffusion. The influence of vacancy clusters on diffusion or segregation processes was not revealed.

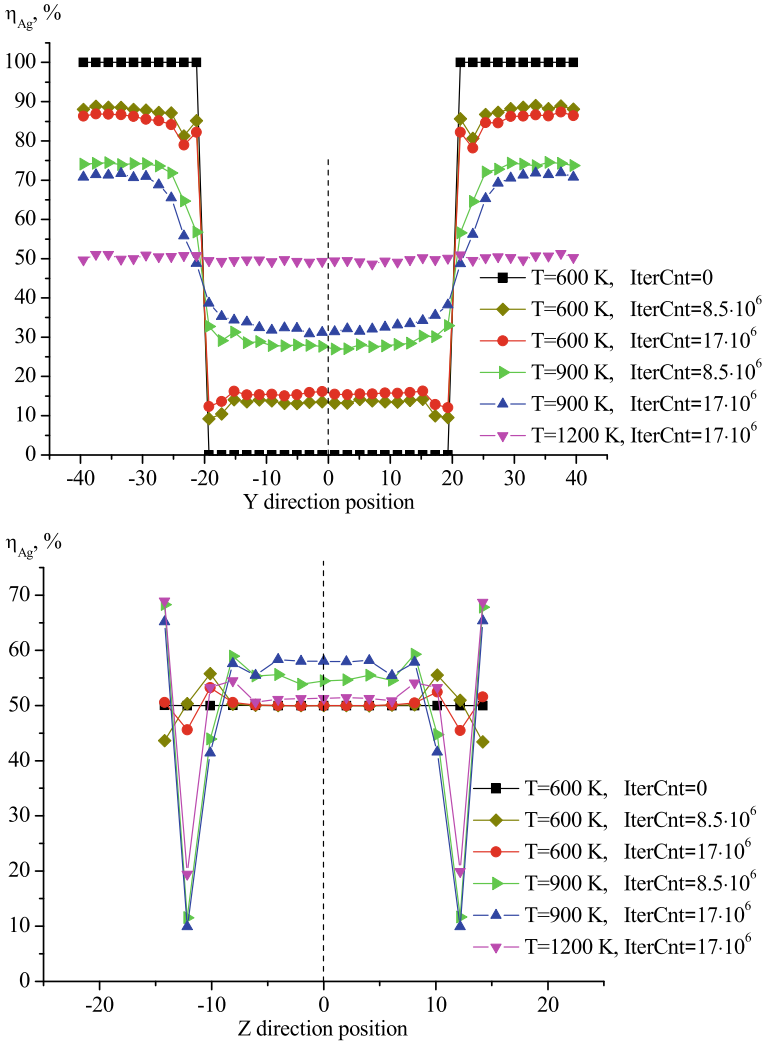


Fig. 5 Distribution of Ag atoms along selected directions for the «sandwich» configuration with the vacancy concentration 8/12000

(4) The final distribution profile of atoms in the Z direction (perpendicular to the surface) is identical in structure for both the «sandwich» configuration and the «random» configurations. At a temperature of less than 1200 K, complete ordering does not occur. In this case, the profile differs only in the width of the equiatomic zone, where the ratio of the number of Ag and Au atoms practically does not change.

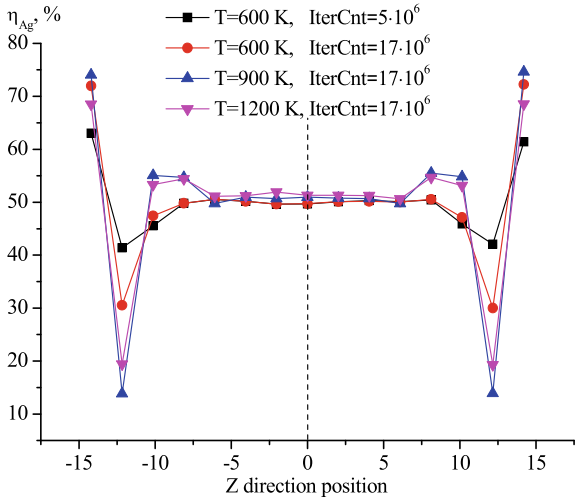


Fig. 6 Distribution of Ag atoms along the Z direction for therelaxed «random» configuration with the vacancy concentration 2/12000

Acknowledgements This work was partially funded by Russian Federation of Basic Research, project number 20-37-70007, by the Ministry of Science and Higher Education of the Russian Federation in the framework of the State Program in the Field of the Research Activity, project number 0817-2020-0007. Stefka Fidanova was supported by the Bulgarian NSF under the grant DFNI-DN 12/5 and by the Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program and co-financed by the European Union through the European structural and Investment funds. LeoneedKirilov and Rossen Mikhov were supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", Ministry of Education and Science—Bulgaria, and by the Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program and co-financed by the European Union through the European structural and Investment funds.

References

1. Abraham, A., Nie, H., Schoenitz, M., et al.: Bimetal Al-Ni nano-powders for energetic formulations. Combust. Flame **173**, 179–186 (2016). https://doi.org/10.1016/j.combustflame.2016.08.015
2. Ali, S., Myasnichenko, V.S., Neyts, E.C.: Size-dependent strain and surface energies of gold nanoclusters. Phys. Chem. Chem. Phys. **18**(2), 792–800 (2016). https://doi.org/10.1039/C5CP06153A
3. Babicheva, R.I., Semenov, A.S., Dmitriev, S.V., Zhou, K.: Effect of grain boundary segregations on martensitic transformation temperatures in NiTi bi-crystals. Lett. Mater. **9**(2), 162–167 (2019). https://doi.org/10.22226/2410-3535-2019-2-162-167
4. Bashkova, D.A., Gafner, Y.Y., Gafner, S.L.: On the prospects of using a phase transition in Ag nanoclusters for information recording processes. Lett. Mater. **9**(4), 382–385 (2019). https://doi.org/10.22226/2410-3535-2019-4-382-385
5. Cleri, F., Rosato, V.: Tight-binding potentials for transition metals and alloys. Phys. Rev. B **48**(1), 22–33 (1993). https://doi.org/10.1103/PhysRevB.48.22

6. Davoodi, J., Mehri, L.: Molecular dynamics simulation of solidification of ag-x%aunanoalloy. Defect Diffus. Forum **312–315**, 143–148 (2011). https://doi.org/10.4028/www.scientific.net/DDF.312-315.143
7. Galashev, A.Y., Rakhmanova, O.R., Kovrov, V.A., Zaikov, Y.P.: Molecular dynamics study of the stability of aluminium coatings on iron. Lett. Mater. **9**(4), 436–441 (2019). https://doi.org/10.22226/2410-3535-2019-4-436-441
8. Hodak, J.H., Henglein, A., Giersig, M., Hartland, G.V.: Laser-induced inter-diffusion in AuAg core-shell nanoparticles. J. Phys. Chem. B **104**(49), 11708–11718 (2000). https://doi.org/10.1021/jp002438r
9. Husic, B.E., Schebarchov, D., Wales, D.J.: Impurity effects on solid—solid transitionsin atomic clusters. NANO **8**(43), 18326–18340 (2016). https://doi.org/10.1039/C6NR06299G
10. Harrison, W.A.: Tight-binding methods. Surf. Sci. **299–300**, 298–310 (1994). https://doi.org/10.1016/0039-6028(94)90662-9
11. Jahangiri, M., Bajgholi, A., Fallah, A., Khodabandeh, A.: Effect of annealing methodand applied stress on aging behavior of copper-aluminum bimetals. J. Ofalloys Compd. **816**,(2019). https://doi.org/10.1016/j.jallcom.2019.152676
12. Julin, J., Napari, I., Vehkamäki, H.: Comparative study on methodology in molecular dynamics simulation of nucleation. J. Chem. Phys. **126**(22), 224517-1–224517-8 (2007). https://doi.org/10.1063/1.2740269
13. Korznikova, E.A., Sharapov, E.A., Khalikov, A.R., Dmitriev, S.V.: Simulation of the binary alloy ordering kinetics during diffusion bonding. Mater. Technol. Des. **1**(1), 58–64 (2019) (in Russian)
14. Li, J.H., Dai, X.D., Liang, S.H., et al.: (2008) Interatomic potentials of the binary transition metal systems and some applications in materials physics. Phys. Rep. **455**(1–3), 1–134 (2008). https://doi.org/10.1016/j.physrep.2007.09.004
15. Li, X.-Y., Zhu, B., Qi, R., Gao, Y.: Real-time simulation of nonequilibriumnano-crystal transformations. Adv. Theory Simul. **2**(1), 1800127-1–1800127-8 (2018). https://doi.org/10.1002/adts.201800127
16. Lobzenko, I.P., Khalikov, A.R., Sharapov, E.A., et al.: Monte-Carlo simulation of the process of diffusion welding of two metals through a gasket. Basic Probl. Mater. Sci. **16**(1), 47–54 (2019). https://doi.org/10.25712/ASTU.1811-1416.2019.01.007
17. Makarov, S.V., Olimov, M.G., Plotnikov, V.A., Hlebutina, S.V., Yakovleva, O.S.: Formation of a diffusion zone at the interface of nickel/aluminum bimetal. News of Altai State University **99**(1), 29–33 (2018). https://doi.org/10.14258/izvasu(2018)1-04
18. Mao, S., Zhang, Y., Li, H., et al.: Writing of nanowiresvia high viscosity-induced nano diffusive layer. J. Mater. Chem. C **5**(45), 11666–11671 (2017). https://doi.org/10.1039/C7TC03962J
19. Myasnichenko, V.S., Starostenkov, M.D.: Formation of fivefold axes in the FCC-metal nanoclusters. Appl. Surf. Sci. **260**, 51–53 (2012). j.apsusc.2012.03.039
20. Myasnichenko, V., Kirilov, L., Mikhov, R., Fidanova, S., Sdobnyakov, N.: Simulatedannealing method for metal nanoparticle structures optimization. Adv. Comput. Ind. Math.: Stud. Comput. Intell. **793**, 277–288 (2019). https://doi.org/10.1007/978-3-319-97277-0_23
21. Myasnichenko, V., Sdobnyakov, N., Kirilov, L., Mikhov, R., Fidanova, S.: Structural instability of gold and bimetallic nanowires using Monte Carlo simulation. In: Recent Advances in Computational Optimization. Studies in Computational Intelligence, vol. 838, pp. 133–145 (2020).https://doi.org/10.1007/978-3-030-22723-4_9
22. Ren, H.S., Ren, X.Y., Xiong, H.P., et al.: Nano-diffusionbonding of Ti2AlNb to Ni-based superalloy. Mater. Charact. **155**, 109813 (2019). https://doi.org/10.1016/j.matchar.2019.109813
23. Samsonov, V.M., Sdobnyakov, N.Y., Bembel, A.G., Sokolov, D.N., Novozhilov, N.V.: Thermodynamic approach to the size dependence of the melting temperatures of films. Bull. Russ. Acad. Sci.: Phys. **78**(8), 733–736 (2014). https://doi.org/10.3103/S1062873814080310
24. Shirzadi, A., Wallach, E.: New method to diffusion bond superalloys. Sci. Technol. Weld. Joining **9**(1), 37–40 (2004). https://doi.org/10.1179/136217104225017125

25. Wu, X., Sun, Y.: Stable structures and potential energy surface of the metallicclusters: Ni, Cu, Ag, Au, Pd, and Pt. J. Nanopart. Res. **19**, 201 (2017). https://doi.org/10.1007/s11051-017-3907-6
26. Yuldasheva, A.R., Nugaeva, N.M.: Magnetoelectric interaction at the interface in thesuperlattices of multiferroic: Monte Carlo study of the phase transitions. Lett. Mater. **9**(3), 354–359 (2019). https://doi.org/10.22226/2410-3535-2019-3-354-359
27. Zakharov, P.B., Markidonov, A.V., Starostenkov, M.D., et al.: Effect of shock post-cascade waves on the interface boundary of bimetals Ni-Al, Ni-Fe. Basic Probl. Mater. Sci. **13**(1), 77–83 (2016) (in Russian)
28. Zaporozhets, T.V., Gusak, A. M., Podolyan, O. M.: Evolution of pores in nanoshells- a competition of direct and inverse Kirkendall effects, Frenkel and Gibbs–Thomson effects: the phenomenological description and computer simulation. Prog. Phys. Metals **13**(1), 1–70 (2012) (in Russian). https://doi.org/10.15407/ufm.13.01.001
29. Zhang, Z., Chen, C., Liu, G., et al.: Enhancement of bonding strength in Ag sinter joining on Au surface finished substrate by increasing Au grain-size. Appl. Surf. Sci. **485**, 468–475 (2019). https://doi.org/10.1016/j.apsusc.2019.04.228
30. Zolnikov, K.P., Kryzhevich, D.S., Korchuganov, A.V.: Atomic mechanisms of highspeedmigration of symmetric tilt grain boundaries in nanocrystalline Ni. Lett. Mater. **9**(2), 197–201 (2019). https://doi.org/10.22226/2410-3535-2019-2-197-201
31. Zorya, I.V., Poletaev, G.M., Rakitin, R.Y., Ilyina, M.A., Starostenkov, M.D.: Interactionof impurity atoms of light elements with self-interstitials in fcc metals. Lett. Mater. **9**(2), 207–211 (2019). https://doi.org/10.22226/2410-3535-2019-2-207-211

On the Problem of Bimetallic Nanostructures Optimization: An Extended Two-Stage Monte Carlo Approach

Rossen Mikhov, Vladimir Myasnichenko, Leoneed Kirilov, Nickolay Sdobnyakov, Pavel Matrenin, Denis Sokolov, and Stefka Fidanova

Abstract In this paper we present an extended version of the two-stage lattice Monte Carlo approach for optimization of bimetallic nanoalloys proposed in (Mikhov R., Myasnichenko V., Kirilov L., Sdobnyakov N., Matrenin P., Sokolov D., Fidanova S. A Two-Stage Monte Carlo Approach for Optimization of Bimetallic Nanostructures. In: Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), September 6–9, 2020. Sofia, Bulgaria, 285–288 (2020)). The two stages consist of simulated annealing on a larger lattice, followed by simulated diffusion. Both constituent algorithms are fairly similar in structure, but their combination was found to give significantly better solutions than simulated annealing alone. We test the proposed approach on a few examples and discuss how to tune the parameters of the algorithms so that they work together optimally.

1 Introduction

The search for stable configurations is a very important research problem in the field of structural characterization of nanoparticles [20, 43, 44]. A configuration is stable when its potential energy is minimal. Thus, the problem can be solved as a global optimization problem. Applying traditional numerical methods is impractical because they need huge amounts of computational resources like time and memory

R. Mikhov · L. Kirilov · S. Fidanova (✉)
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Sofia, Bulgaria

L. Kirilov
e-mail: l_kirilov_8@abv.bg

V. Myasnichenko · N. Sdobnyakov · D. Sokolov
Tver State University, Tver, Russia

P. Matrenin
Novosibirsk State Technical University, Novosibirsk, Russia

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_12

[8]. Therefore, the global minimum has to be approximated using time-efficient optimization strategies (metaheuristics).

A lot of methods are available for the prediction of nanoparticle structures [10]. An implementation of the basin-hopping algorithm is studied in [32] for the optimization of the potential energy of nanoclusters, combining two different types of moves, namely shape-changing moves and exchange moves. The approach is demonstrated on nanoalloys of 400 atoms composed by Au and Rh, or by Au and Cu. A review of the new applications of GAs in materials science and in related fields (solid state physics and chemistry, crystallography, production and engineering) is presented in [29] with representative examples. In [31], a global optimization problem of determining the n-atom cluster configuration that yields the minimum Lennard–Jones potential energy is studied by a genetic algorithm combined with a stochastic search procedure on icosahedral lattices. In [8], the physical aspects of the global optimization of the geometry of atomic clusters are studied. Namely, the structural principles that determine the nature of the lowest energy structure, the physical reasons why some clusters are especially difficult to optimize, and how the basin-hopping transformation of the potential energy surface enables these difficult clusters to be optimized.

An example of the description of the chemical ordering procedure in bimetallic nanoparticles can be seen in the results of [16]. In [16], it is established that chemical ordering in bimetallic nanocrystallites can be efficiently determined by density-functional calculations with the help of topological energy expressions. It is shown that this enables one to structurally characterize bimetallic nanoparticles of less regular shapes than previously studied magic-type Pd-Au particles of different shapes. On the other hand [38], a structural optimization procedure can be realized for Au–Pd bimetallic nanoparticles by using an improved particle swarm optimization method. This illustrates why further application of numerical algorithms to implement the procedure of structural optimization in bimetallic nanoparticles is an urgent and rather nontrivial problem.

A combination of simulation tools for thermodynamic properties and stability of nanosystems is proposed in [3]—mainly parallel Monte Carlo algorithms for icosahedral, multilayer Pd–Pt clusters. The model is on a 3D cubic lattice. In [4], grand and semigrand canonical global optimization approaches are presented, using basin-hopping with an acceptance criterion based on the local contribution of each potential energy minimum to the (semi) grand potential. Practical details regarding the implementation of the basin-hopping method are also given in relation to Monte Carlo moves that change the system size.

In [7], the chemical ordering in "magic-number" Pd-Ir nanoalloys is studied. The density functional theory is compared with the results of the free energy concentration expansion method. In [9], the problem of stable structures of alloy nanoparticles is investigated. A two-step search strategy is proposed. The first strategy is based on extensive global optimization search and is combined with an empirical potential with density-functional local relaxation. The structure and thermodynamics of Cu–Ni nanoalloys is studied in [26]. An atomic model is described in the framework of a potential based on the second-moment approximation of the tight binding potential.

In [27], a novel structure for free Co–Pt nanoalloys is developed. Three computational methodologies have been combined. The energetic stability of the novel structure has been checked. The structure and energetics of Pd–Pt nanoalloys are studied in [28]. The model is based on the second-moment approximation to tight binding theory. To solve the problem the authors apply a genetic algorithm. Schebarchov and Wales [36] study the problem of structure predicting of multi-component systems. The system is represented as a generalized graph. They apply a Kernighan and Lin heuristic procedure to find locally optimal partitions of an arbitrary graph. The same problem is studied in [37]. A local optimization technique combined with multiple local-neighborhood searches is applied. In [39], a parallel modification of the Birmingham cluster genetic algorithm is proposed for global optimization of nanoalloy clusters using a pool strategy. The method is illustrated for global optimization of the Au10Pd10 cluster using the Gupta potential. The structure of different AuCu clusters is studied in [40] by means of a Parallel Excitable Walkers algorithm and molecular dynamics. The basin-hopping Monte Carlo algorithm was modified to determine a global minimum structure in Ag and AgPd nanoclusters [14]. For a pure metallic silver nanocluster, the newly developed quadratic basin-hopping Monte Carlo algorithm is more efficient than the standard basin-hopping Monte Carlo algorithm. For a bimetallic AgPd nanocluster, the new algorithm succeeds in finding the global minimum structure even though the standard basin-hopping Monte Carlo algorithm fails to achieve it. It is important that such approach as the formation energy machine learning model [5] can be used to predict the stable metal element distribution in the nanoparticles via Monte Carlo simulations.

As it is known, atom ordering in the process of forming nanoalloys is one of the factors defining the structure and properties of the materials. In particular, in the case of bimetallic nanostructures, atomic diffusion processes can strongly affect their properties, because they are often also determined precisely by the mutual arrangement of the alloy components. Knowing the mechanism of this process helps to obtain materials with preset properties. At present this kind of problem is extensively studied [18]. An example is welding due to the mutual diffusion at the atomic level of the surfaces of parts to be welded (diffusion welding)—[15]. At present, the many-body interatomic interaction potentials for metals and their binary compounds are well developed, which allows us to simulate the processes of diffusion and phase transitions in bimetallic nanosystems with a high degree of accuracy—[13, 17]. Surface diffusion and structural relaxation processes were studied in mono-metallic gold and copper nanoparticles up to 10,179 atoms in size—[1], in bimetallic Cu–Au particles—[23], and in Au–Ag particles—[24]. In [19], the formation of a diffusion zone at the Ni/Al interface is studied. The patterns of the formation of the diffusion zone are studied on samples with different time intervals of annealing. Ren et al. [33] study nano-diffusion bonding of Ti_2AlNb to Ni-based superalloy. Raising bonding temperature could enhance metallurgical bonding and improve bonding quality. Babicheva et al. [2] investigate the effect of grain boundary segregations on martensitic transformation temperatures in NiTi bi-crystals. The segregation is simulated by replacing the monoatomic Ti or Ni layer in the grain boundary with a Ni or Ti layer, respectively. At the nanoscale, appropriate size effects should be taken into account, in

particular, at the interface between components [34, 35]. Moreover, mutual diffusion in binary nanocrystalline systems consisting of two chemical elements is most often provided by a vacancy mechanism [45]. In particular, the work of [12] can be noted. This study investigates the effect of aging method and the presence or absence of external stress on the aging behavior of Cu/Al diffusion couples. Hodak et al. [11] demonstrate the synthesis of bimetallic Au(core)Ag(shell) and Ag(core)Au(shell) nanoparticles using radiation chemistry. It is noted that interdiffusion of Ag and Au in the core–shell nanoparticles also occurs at temperatures that are much lower than the expected melting point. Zhang et al. [46] used SEM, EDS and XRD characterizations to analyze improvement of bonding strength, diffusion behavior between Au and Ag, and structure of the Au layer.

2 The Basic Algorithms

We investigate the stable configurations of bimetallic nanostructures using Monte Carlo simulation. This is an extended version of the two-stage approach we proposed in [21]. The optimization is performed on a lattice, combining two Monte Carlo algorithms. The energy of the system is given by the multi-particle tight-binding potential of Cleri–Rosato [6], having the following form:

$$\begin{aligned} E &= \sum_i \sum_{j \neq i} E_{ij,ab} - \sum_i \sqrt{\sum_{j \neq i} B_{ij,ab}} \\ E_{ij,ab} &= A_{ab} \exp\left(-p_{ab}\left(\frac{r_{ij}}{r_{0,ab}} - 1\right)\right) \\ B_{ij,ab} &= \xi_{ab}^2 \exp\left(-2q_{ab}\left(\frac{r_{ij}}{r_{0,ab}} - 1\right)\right), \end{aligned} \tag{1}$$

where i ranges over all atoms; j ranges over all atoms other than i but within distance R_{cut} from i; a and b represent the species of the atoms i and j; $E_{ij,ab}$ and $B_{ij,ab}$ are the repulsive and binding components of the potential due to the atom pair (i, j); r_{ij} is the distance between the atoms; $r_{0,ab}$, A_{ab}, p_{ab}, ξ_{ab}, q_{ab} are constants particular to the chemical elements under consideration. We use a value for R_{cut} corresponding to five coordination spheres, beyond which the interaction is assumed to be zero.

The following subsections give a brief overview of the constituent basic algorithms, before describing the full combined method in Sect. 3. After that, in Sect. 4, we present data from numerical experimentation to demonstrate the advantages of the approach, and compare different ways for tuning the parameters.

2.1 The Wide-Lattice Monte Carlo Algorithm

The first algorithm, which we will call the "wide-lattice" Monte Carlo, is specified in [24]. It starts by placing the atoms at random on a lattice several times larger than the total number of atoms. At each iteration, one atom and one neighboring empty node are chosen at random. If the potential energy would decrease by the atom moving into the empty node, the jump is performed unconditionally. Otherwise, the jump may still be performed, with a probability given as:

$$P = \exp(-\Delta E/kT), \tag{2}$$

where ΔE is the energy difference of the configurations and T is the current temperature of the system. The iteration ends either with or without a jump.

The temperature is set high at the beginning, and then gradually decreases as the algorithm proceeds. We use a linear formula for the cooling, subtracting a small amount once every several thousand iterations. The algorithm ends when the temperature reaches 1 K.

The appropriate initial temperature strongly depends on the size and type of lattice used, as well as on the size and chemical composition of the nanostructure [22], therefore it is best determined experimentally.

Due to its simplicity and its particular form, this algorithm lends itself to a highly optimized computer implementation. It can run for billions of iterations within minutes on a standard personal computer.

2.2 The Diffusion Algorithm

The second algorithm, which we will call the "diffusion," is specified in [25]. It runs on a lattice filled with atoms of two different kinds, plus a small number of empty nodes (~4 for a 200-atom structure). At each iteration, one empty node is chosen at random, and the iteration always ends with a jump of a neighboring atom into that empty node. Which atom jumps is determined by calculating (2) separately for each candidate, and picking a random number in the interval from zero to the sum of all P s.

Note that the term *neighboring atom* is defined here as being within a radius of three coordination spheres. This is different from the wide-lattice Monte Carlo algorithm above, where nodes are neighbors only within one coordination sphere. This difference is due to the scarcity of the empty nodes during diffusion, and we have verified experimentally that three coordination spheres seem to be optimal for this purpose.

Temperature is managed similarly to the wide-lattice Monte Carlo algorithm.

The running time, while slower than the wide-lattice Monte Carlo, is still on the order of millions of iterations per minute.

2.3 Relaxation with Molecular Dynamics

It is existing practice [24] that after all Monte Carlo simulation is completed, an additional step of relaxation with molecular dynamics (MD) can be performed to further improve the energy of the system. In an MD simulation, the time trajectories of all atoms in the system are followed through space by integrating Newton's equations of motion. Forces are derived as the negative gradient from the interatomic potential.

In this work, the so-called modified Verlet velocity algorithm [41, 42] is used, where positions, velocities and accelerations at iteration $t + \Delta t$ are calculated as follows:

$$\vec{r}(t + \Delta t) = \vec{r}(t) + \vec{v}(t)\Delta t + \frac{1}{2}\vec{a}(t)(\Delta t)^2 \tag{3}$$

where $U(\vec{r}(t))$ and m are the potential energy and the mass of the particle, respectively.

The advantage of the modified Verlet velocity algorithm is that it is self-starting:

$$\vec{v}(t + \Delta t) = \vec{v}\left(t + \frac{1}{2}\Delta t\right) + \frac{1}{2}\vec{a}(t + \Delta t)\Delta t. \tag{4}$$

The sequential solution of (3)–(4) constitutes one elementary modeling step.

In the molecular dynamics method, no probabilistic considerations are involved, with the exception of the initial conditions. When starting from predetermined positions of the particles, only velocities are initially chosen randomly. The absolute values of the velocities should be determined in such a way that the total kinetic energy of the system must correspond to the set temperature.

It is believed [30] that the problem of choosing the initial velocities in molecular dynamics modeling is not fundamental, since the velocities corresponding to a given temperature are established in just a few steps of the evolution of the system. On the one hand, this is true. On the other hand, in accordance with our results, in the transition from modeling a massive phase using periodic boundary conditions to modeling small objects, an adequate choice of the initial state of the system becomes very important. At the same time, it should be noted that the choice of both zero initial velocities (zero absolute temperature) and an arbitrary high temperature is possible if the simulation algorithm presupposes preliminary relaxation of the system to an equilibrium state, at which the velocity distribution becomes Maxwellian.

3 The Combined Method

The starting point of this research is the observation that combining the two Monte Carlo algorithms above may produce better solutions than just a single-staged approach. The proposed combined method has the following steps:

Step 1: Parameter tuning. Repeatedly run the wide-lattice Monte Carlo algorithm from random initial configurations, for N_1 (~40 million) iterations at each trial, to determine the optimal initial temperature T_2 and scaling factor (along each of the x, y, and z axes) for the lattice. Due to the lower number of iterations used, each trial completes quickly to save time for the more important following steps.

Step 2: Shape fixing. Using the best values from Step 1 for the initial temperature T_2 and for the scaling factors, repeatedly run the wide-lattice Monte Carlo algorithm afresh from random initial configurations, for N_2 (~1 billion) iterations at each trial. Note that the cooling speed here is much slower than that of Step 1: since we use a linear formula for cooling, the cooling speed is basically the initial temperature divided by the number of iterations.

For Step 2, initially set up the nanoalloy to have α (~2) extra atoms of each type. After Step 2 completes, delete all empty nodes, then convert α randomly selected atoms of each type into empty nodes, to use as vacancies during diffusion. The goal of this step is to obtain the advantageous geometric shapes appropriate for the nanoparticle (3D), film (2D) or wire (1D) under consideration on the given lattice. The resulting configurations also have a somewhat low-energy ordering of the atoms, to be further improved by diffusion.

Step 3: Parameter tuning for diffusion. Repeatedly run the diffusion algorithm on the results of Step 2, for N_3 (~10 million) iterations at each trial, to determine the optimal initial temperature T_4 for diffusion. Again, a lower number of iterations is used during parameter tuning, so that the trials complete quickly.

Step 4: Diffusion. Using the best value from Step 3 for the initial temperature T_4, run the diffusion algorithm afresh on the results of Step 2, once per configuration, for N_4 (~100 million) iterations at each trial.

Step 5: Relaxation. For each resulting configuration, use the molecular dynamics method to further improve the energy of the system. Select the best solution as the final result. MD simulations are carried out in the canonical ensemble at 0.1 K, with the time step of 0.5 fs.

4 Verification

We did a number of tests to verify that the proposed method is a significant improvement over simpler approaches, and to adjust the values of the parameters.

Unless stated otherwise, all trials were performed for a 200-atom AuAg nanoparticle (gold and silver in 1:1 proportion), on a 309-node lattice with a twinned bipyramid shape. One example configuration of this particle is illustrated on Fig. 1. It

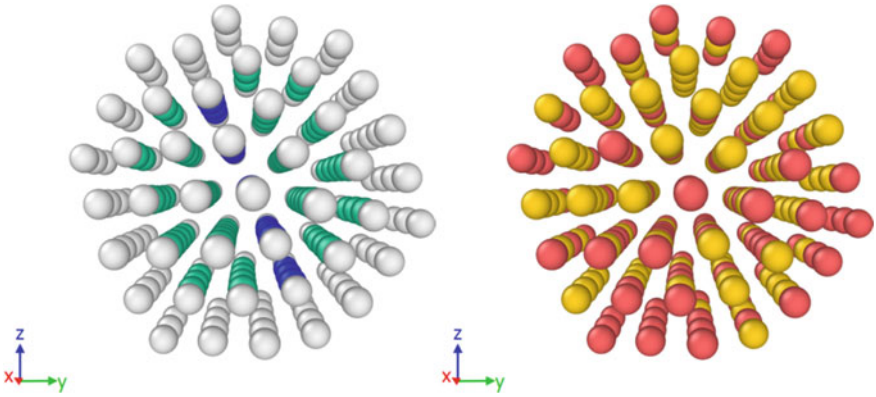


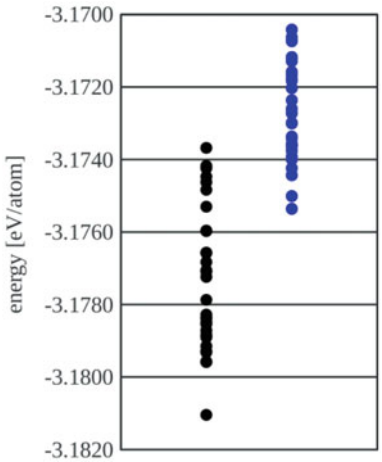
Fig. 1 One example configuration of Au100Ag100 (left: blue—fcc atoms, green—hcp atoms, grey—unknown atoms; right: yellow—Au atoms, red—Ag atoms)

is expected that for other chemical compositions and other lattices, the same results would hold, although probably not with the same particular values of the parameters. This has been observed in our preliminary testing.

In order to simplify the discourse, instead of relaxation with MD (Step 5), all comparisons of final solutions below are done after applying an additional round of scaling of the lattice (with separate factors for the x, y, and z axes). The tests were run on an Intel Core i3-9300 3.70 GHz CPU, utilizing one core per trial. The running time was approximately 0.6 s per 1 million iterations of the wide-lattice algorithm, and 12 s per 1 million iterations of the diffusion algorithm. The total running time for Steps 2 + 4 of a typical trial (e.g. $N_2 = 1226$ million, $N_4 = 128$ million) was 37 min. The algorithms used are not memory-intensive.

First, we compare the proposed two-stage method against using only the wide-lattice algorithm. In Fig. 2, the combined method (Steps 1 to 4) is compared against running only Steps 1 + 2, but for a larger number of iterations, and with = 0. The number of iterations is chosen so that the total wall-clock running time is the same in both cases. The combined method gives clearly better solutions, which is the main result of this research.

Next, we address the problem of time allocation between the different stages. Since Steps 1 and 3 take only a small constant amount of time, we concentrate on Steps 2 and 4. In Fig. 3, the results of using different values for N_2 and N_4 are compared. In order for the comparison to be fair, the values are chosen so that the total running time for each trial would be approximately the same, and then the real running time is plotted as a proportion: the time t_2 spent at Step 2, over the time t_{2+4} spent at Steps 2 + 4. It is not shown on the figure, but we note that the total average running time t_{2+4} for each column was indeed approximately the same (within ±5%), with the third (black) column actually taking a little less (−2%) time, even though clearly it is producing the best solutions. This gives us grounds to conclude that the values $N_2 = 1226$ million and $N_4 = 128$ million are the most adequate for our setting, and



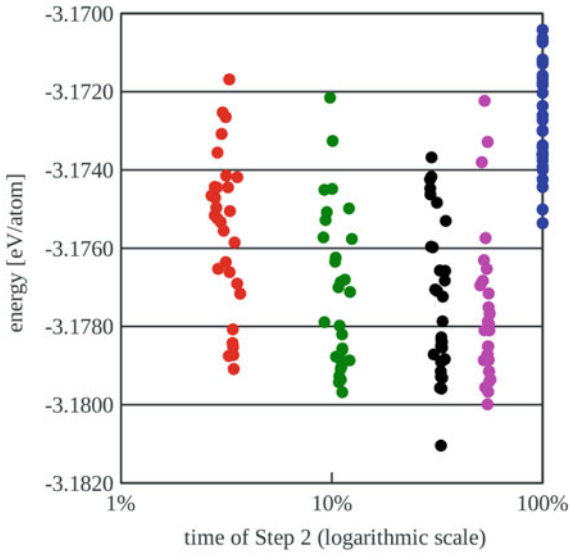
left side (black) dots: 30 trials	right side (blue) dots: 30 trials
combined method	wide-lattice algorithm only
Step 1: $N_1 = 40$ million	Step 1: $N_1 = 40$ million
Step 2: $N_2 = 1226$ million; $T_2 = 2500$ K; $\alpha = 2$	Step 2: $N_2 = 4$ billion; $T_2 = 2500$ K; $\alpha = 0$
Step 3: $N_3 = 17$ million	
Step 4: $N_4 = 128$ million; $T_4 = 1400$ K	(so that total running time is the same)

Fig. 2 Comparing the combined method (Steps 1–4) against using only the wide-lattice simulated annealing algorithm (Steps 1 + 2)

that a good strategy in general might be to allocate 30% of the running time to Step 2 and 70% to Step 4.

The next series of experiments aims to verify that the parameter tuning approach of Steps 1 and 3 is sound. In other words, that the optimal initial temperatures T_2 and T_4 determined at the high cooling speeds of Steps 1 and 3 would still be optimal when the algorithms are run at the much lower cooling speeds of Steps 2 and 4. From Fig. 4, it can be read out that the initial temperatures giving best results are $T_2 = 2500$ K and $T_4 = 1400$ K. This is basically true irrespective of cooling speed, though there is a very slight bias towards lower initial temperature with the slower cooling. We note that for this lattice, the differences between initial temperatures seem to be small, to the point that any temperature from the chosen range may be adequate. With larger lattices and more atoms, however, the influence of the initial temperature is more dramatic [22], and in that case the advantage of knowing the best initial temperature before running the main algorithm is also expected to be more substantial.

The last series of tests on this lattice evaluate whether the parameter tuning approach of Step 1 is advantageous. That is, whether the solutions by the combined method are better than if we omitted Step 1 and used alternative ways to tune the parameters. In Fig. 5, comparing the left (black) column with the middle (red) column,

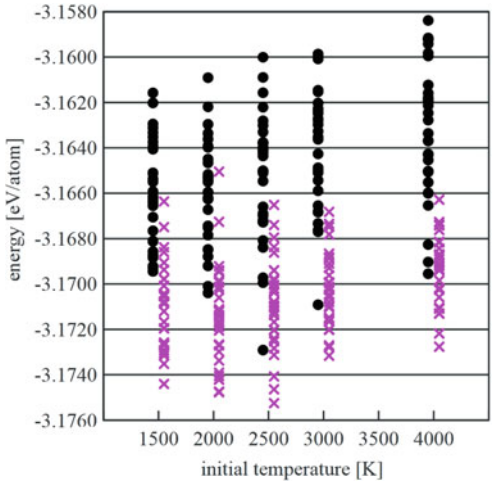


first (red), second (green), third (black), fourth (pink) columns of dots: 30 trials each
combined method Step 1: $N_1 = 40$ million Step 2: $N_2 = 125$ million, 400 million, 1226 million, 2052 million; $T_2 = 2500$ K; $\alpha = 2$ Step 3: $N_3 = 17$ million Step 4: $N_4 = 184$ million, 170 million, 128 million, 86 million; $T_4 = 1400$ K
rightmost (blue) column of dots: 30 trials
wide-lattice algorithm only; the same as in Fig. 2, shown here for reference

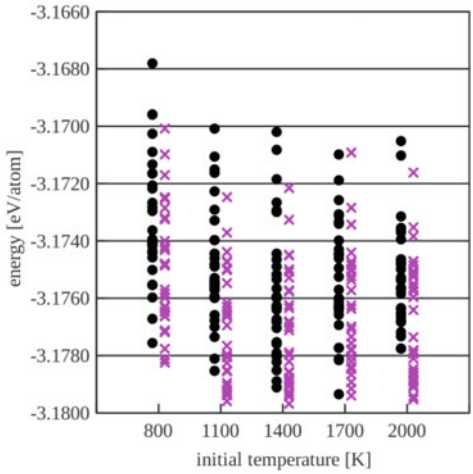
Fig. 3 Comparing different allocations of time for Steps 2 and 4, i.e. different proportions of time spent at Step 2, over the total time spent at Steps 2 + 4 for that trial

it can be seen that there is not much difference between the combined method and the variant omitting Step 1 but running Step 2 for more iterations to compensate (and since the best initial temperature is assumed unknown, from a random temperature in the range of 1000–4000 K). This corresponds to the fact that the influence of the initial temperature is small, as was already observed in Fig. 4. It remains open whether choosing the initial temperature at random is adequate for larger numbers of atoms, where the temperature effects are amplified.

In Fig. 5, the right (green) column shows the case of an alternative way of determining the lattice scaling factors to be applied before diffusion: instead of performing the scaling after Step 1 and using that for Steps 2–4, omit Step 1 and perform the scaling after Step 2. Comparing these solutions to the left (black) column, we see that the alternative approach is not any worse. It may even look like it is slightly better, but this cannot be determined from our data because the experiment was designed as a one-sided test. (The alternative (green) trials run Step 2 for 10% more iterations

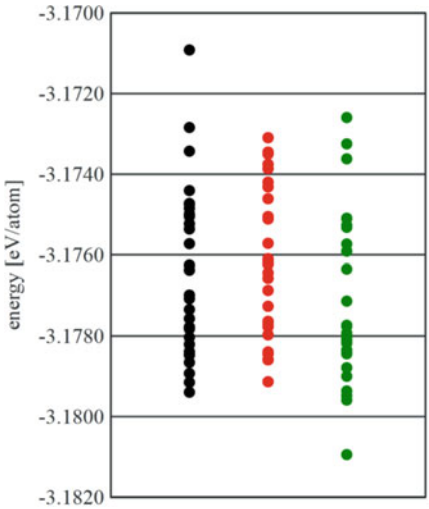


Wide-lattice algorithm. **For each initial temperature:**	
left side (black dots): 30 trials	**right side (pink crosses) 30 trials**
fast cooling (40 million iterations)	slow cooling (400 million iterations)



Diffusion algorithm. **For each initial temperature:**	
left side (black dots): 30 trials	**right side (pink crosses) 30 trials**
fast cooling (17 million iterations)	slow cooling (170 million iterations)

Fig. 4 Comparing the optimal initial temperature of each algorithm for fast cooling (such as in Steps 1 and 3) versus slow cooling (such as in Steps 2 and 4). The dots and crosses correspond to identical temperatures but are displaced slightly to the left (resp. to the right) for clarity



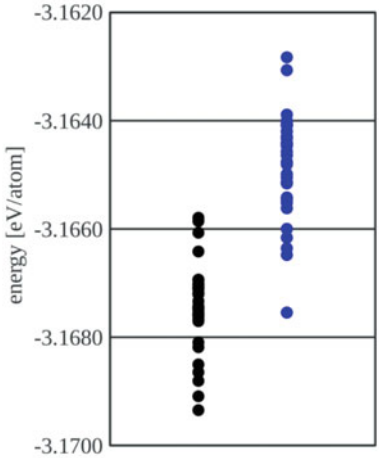
left side (black) dots: 30 trials	**middle (red) dots: 30 trials**	**right side (green) dots: 30 trials**
combined method	alternative 1	alternative 2
Step 1: $N_1 = 40$ million	random T_2	scaling of the lattice after Step 2
Step 2: $N_2 = 400$ million; $T_2 = 2500$ K	Step 2: $N_2 = 440$ million; $T_2 \in [1000,4000]$ K	Step 2: $N_2 = 440$ million; $T_2 = 2500$ K
Step 4: $N_4 = 170$ million; $T_2 = 1700$ K	Step 4: $N_4 = 170$ million; $T_2 = 1700$ K	Step 4: $N_4 = 170$ million; $T_2 = 1700$ K

Fig. 5 Influence on the final results of using Step 1 to tune the parameters: the initial temperature (comparing left vs. middle columns), and the lattice scaling factor (comparing left vs. right columns)

than the left (black) trials while still starting from the known best initial temperature $T_2 = 2500$ K, which gives them an unfair advantage.)

Together, the tests of Figs. 4, 5 imply that the parameter tuning approach of Steps 1 and 3 is adequate but not necessarily essential. If there is enough prior information available about the behavior of the lattice at hand (for example, from other experiments under a similar setting), Steps 1 and 3 may be omitted.

As a final example, the effect of the proposed combined method is verified on another lattice. Figure 6 shows the results for the same AuAg nanoparticle on a 309-node icosahedral lattice. The advantage of the left (black) column is clear, similarly to Fig. 2.



left side (black) dots: 30 trials	right side (blue) dots: 30 trials
combined method	wide-lattice algorithm only
Step 1: $N_1 = 40$ million	Step 1: $N_1 = 40$ million
Step 2: $N_2 = 1200$ million; $T_2 = 2500$ K; $\alpha = 2$	Step 2: $N_2 = 4$ billion; $T_2 = 2500$ K; $\alpha = 0$
Step 3: $N_3 = 13$ million	
Step 4: $N_4 = 130$ million; $T_4 = 1400$ K	(so that total running time is the same)

Fig. 6 The results for an icosahedral lattice: comparing the combined method (Steps 1 to 4) against using only the wide-lattice simulated annealing algorithm (Steps 1 + 2)

5 Conclusion

The computer modeling of bimetallic nanosystems is more complicated from the point of view of the interpretation of the results, compared to one component systems. This problem is of exceptional importance when studying the properties of nanomaterials.

We have proposed an extended version of a lattice Monte Carlo method for optimization of bimetallic nanoalloys, combining two previous algorithms: a wide-lattice simulated annealing algorithm and a simulated diffusion algorithm. The combined method was verified to give significantly better solutions than using only wide-lattice simulated annealing.

We have discussed several aspects of the combined method, including time allocation between the two major stages, and ways to tune the algorithm parameters.

For parameter tuning, we have proposed one particular approach, consisting of two additional tuning steps: fast-cooling trial runs to determine the appropriate temperatures/scaling factors before each of the two respective major stages. The soundness of this approach was verified with numerical experiments, and the results were compared to the investigated alternatives.

There are other important parameters to be considered that we have not discussed here. In particular, with simulated annealing, it is often advantageous to use a more complicated cooling schedule, as well as the occasional use of restarting. These considerations are expected to become more important when a large number of atoms are involved. Further research will be needed to determine the optimal strategy for particle sizes where this becomes relevant.

Acknowledgements This work was partially funded by Russian Federation of Basic Research, project number 20-37-70007, by the Ministry of Science and Higher Education of the Russian Federation in the framework of the State Program in the Field of the Research Activity, project number 0817-2020-0007. Stefka Fidanova was supported by the Bulgarian NSF under the grant DFNI-DN 12/5 and by the Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program and co-financed by the European Union through the European structural and Investment funds. Leoneed Kirilov and Rossen Mikhov were supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", Ministry of Education and Science – Bulgaria, and by the Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program and co-financed by the European Union through the European structural and Investment funds.

References

1. Ali, S., Myasnichenko, V.S., Neyts, E.C.: Size-dependent strain and surface energies of gold nanoclusters. Phys. Chem. Chem. Phys. **18**(2), 792–800 (2016)
2. Babicheva, R.I., Semenov, A.S., Dmitriev, S.V., Zhou, K.: Effect of grain boundary segregations on martensitic transformation temperatures in NiTi bi-crystals. Lett. Mater. **9**(2), 162–167 (2019)
3. Calvo, F.: Solid-solution precursor to melting in onion-ring Pd-Pt nanoclusters: a case of second-order-like phase change? Faraday Discuss. **138**, 75–88 (2008)
4. Calvo, F., Schebarchov, D., Wales, D.J.: Grand and semigrand canonical basin-hopping. J. Chem. Theory Comput. **12**(2), 902–909 (2015)
5. Chen, C., Zuo, Y., Ye, W., et al.: Critical review of machine learning of energy materials. Adv. Energy Mater. **10**(8), 1903242-1–1903242-36 (2020)
6. Cleri, F., Rosato, V.: Tight-binding potentials for transition metals and alloys. Phys. Rev. B **48**(1), 22–33 (1993)
7. Davis, J., Johnston, R., Rubinovich, L., Polak, M.: Comparative modelling of chemical ordering in palladium-iridium nanoalloys. J. Chem. Phys. **141**, 224307-1–224307-7 (2014)
8. Doye, J.P.K.: Physical perspectives on the global optimization of atomic clusters. In: Pintér, J.D. (ed.) Global Optimization. Nonconvex Optimization and Its Applications, vol. 85. Springer, Boston, MA, pp. 103–139 (2006)
9. Ferrando, R., Fortunelli, A., Johnston, R.: Searching for the optimum structures of alloy nanoclusters. Phys. Chem. Chem. Phys. **10**, 640–649 (2008)
10. Gelfand, S.B., Mitter, S.K.: Metropolis-type annealing algorithms for global optimization in {R}ˆd. SIAM J. Control. Optim. **31**(1), 111–131 (1993)
11. Hodak, J.H., Henglein, A., Giersig, M., Hartland, G.V.: Laser-induced inter-diffusion in AuAg core-shell nanoparticles. J. Phys. Chem. B **104**(49), 11708–11718 (2000)
12. Jahangiri, M., Bajgholi, A., Fallah, A., Khodabandeh, A.: Effect of annealing method and applied stress on aging behavior of copper-aluminum bimetals. J. Alloy. Compd. **816**, 152676 (2019)

13. Julin, J., Napari, I., Vehkamäki, H.: Comparative study on methodology in molecular dynamics simulation of nucleation. J. Chem. Phys. **126**(22), 224517-1–224517-8 (2007)
14. Kim, H.G., Choi, S.K., Lee, H.M.: New algorithm in the basin hopping Monte Carlo to find the global minimum structure of unary and binary metallic nanoclusters. J. Chem. Phys. **128**(14) 144702-1–144702-4 (2008)
15. Korznikova, E.A., Sharapov, E.A., Khalikov, A.R., Dmitriev, S.V.: Simulation of the binary alloy ordering kinetics during diffusion bonding. Mater. Technol. Des. **1**(1), 58–64 (2019). (in Russian)
16. Kovács, G., Kozlov, S.M., Neyman, K.M.: Versatile optimization of chemical ordering in bimetallic nanoparticles. J. Phys. Chem. C **121**(20), 10803–10808 (2017)
17. Li, J.H., Dai, X.D., Liang, S.H., et al.: Interatomic potentials of the binary transition metal systems and some applications in materials physics. Phys. Rep. **455**(1–3), 1–134 (2008)
18. Li, X.-Y., Zhu, B., Qi, R., Gao, Y.: Real-time simulation of nonequilibrium nanocrystal transformations. Adv. Theory Simul. **2**(1), 1800127-1–1800127-8 (2018)
19. Makarov, S.V., Olimov, M.G., Plotnikov, V.A., Hlebutina, S.V., Yakovleva, O.S.: Formation of a diffusion zone at the interface of nickel/aluminum bimetal. News of Altai State University **99**(1), 29–33 (2018). (in Russian)
20. Michaelian, K., Rendón, N., Garzón, I.L.: Structure and energetics of Ni, Ag, and Au nanoclusters. Phys. Rev. B **60**(3), 2000–2010 (1999)
21. Mikhov, R., Myasnichenko, V., Kirilov, L., Sdobnyakov, N., Matrenin, P., Sokolov, D., Fidanova, S.: A two-stage Monte Carlo approach for optimization of bimetallic nanostructures. In: Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS). Sofia, Bulgaria, pp. 285–288 (2020), 6–9 Sept 2020
22. Mikhov, R., Myasnichenko, V., Fidanova, S., Kirilov, L., Sdobnyakov, N.: Influence of the temperature on simulated annealing method for metal nanoparticle structures optimization. In: Advanced Computing in Industrial Mathematics: 13th Annual Meeting of the Bulgarian Section of SIAM. Sofia, Bulgaria, Springer, in press, 18–20 Dec 2018
23. Myasnichenko, V.S., Starostenkov, M.D.: Formation of fivefold axes in the FCC-metal nanoclusters. Appl. Surf. Sci. **260**, 51–53 (2012)
24. Myasnichenko, V., Sdobnyakov, N., Kirilov, L., Mikhov, R., Fidanova, S.: Structural instability of gold and bimetallic nanowires using Monte Carlo simulation. In: Recent Advances in Computational Optimization: Results of the Workshop on Computational Optimization and Numerical Search and Optimization 2018. Studies of Computational Intelligence, vol. 902, pp. 133–145. Springer, Berlin (2020)
25. Myasnichenko, V., Mikhov, R., Kirilov, L., Sdobnykov, N., Sokolov, D., Fidanova, S.: Simulation of diffusion processes in bimetallic nanofilms. In: Recent Advances in Computational Optimization: Results of Workshop on Computational Optimization WCO20, Studies of computational intelligence. Springer, (accepted)
26. Panizon, E., Olmos-Asar, J., Peressi, M., Ferrando, R.: The study of the structure and thermodynamics of CuNi nanoalloys using a new DFT-fitted atomistic potential. Phys. Chem. Chem. Phys. **17**, 28068–28075 (2015)
27. Parsina, I., DiPaola, C., Baletto, F.: A novel structural motif for free CoPt nanoalloys. Nanoscale **4**, 1160–1166 (2012)
28. Paz-Borbon, L., Mortimer-Jones, T., Johnston, R., Posada-Amarillas, A., et al.: Structures and energetics of 98 atom Pd-Pt nanoalloys: potential stability of the Leary tetrahedron for bimetallic nanoparticles. Phys. Chem. Chem. Phys. **9**, 5202–5208 (2007)
29. Paszkowicz, W.: Genetic algorithms, a nature-inspired tool: a survey of applications in materials science and related fields: Part II. Mater. Manuf. Process. **28**, 708–725 (2013)
30. Polukhin, V.A., Vatolin, N.A.: Simulation of Amorphous Metals, p. 285. Nauka, Moscow (1985)
31. Romero, D., Barrón, C., Gómez, S.: The optimal geometry of Lennard-Jones clusters: 148–309. Comput. Phys. Commun. **123**, 87–96 (1999)
32. Rossi, G., Ferrando, R.: Combining shape-changing with exchange moves in the optimization of nanoalloys. Comput. Theor. Chem. **1107**, 66–73 (2017)

33. Ren, H.S., Ren, X.Y., Xiong, H.P., et al.: Nano-diffusion bonding of Ti2AlNb to Ni-based superalloy. Mater. Charact. **155**, 109813 (2019)
34. Samsonov, V.M., Sdobnyakov, N.Yu., Bembel, A.G., Sokolov, D.N., Novozhilov, N.V.: Size dependence of the melting temperature of metallic films: two possible scenarios. J. Nano-Electron. Phys. **5**(4), 04005-1–04005-3 (2013)
35. Samsonov, V.M., Sdobnyakov, N.Y., Bembel, A.G., Sokolov, D.N., Novozhilov, N.V.: Thermodynamic approach to the size dependence of the melting temperatures of films. Bull. Russ. Acad. Sci.: Phys. **78**(8), 733–736 (2014)
36. Schebarchov, D., Wales, D.: A new paradigm for structure prediction in multicomponent systems. J. Chem. Phys. **139**(22), 221101 (2013)
37. Schebarchov, D., Wales, D.: Quasi-combinatorial energy landscapes for nanoalloy structure optimization. Phys. Chem. Chem. Phys. **17**, 28331–28338 (2015)
38. Shao, G.-F., Zhu, M., Shangguan, Y.L., et al.: Structural optimization of Au–Pd bimetallic nanoparticles with improved particle swarm optimization method. Chin. Phys. B **26**(6), 063601 (2017)
39. Shayeghi, A., Götz, D., Davis, J.B.A., Schäfer, R., Johnston, R.L.: Pool-BCGA: a parallelised generation-free genetic algorithm for the ab initio global optimisation of nanoalloy clusters. Phys. Chem. Chem. Phys. **17**, 2104–2112 (2015)
40. Toai, T.J., Rossi, G., Ferrando, R.: Global optimisation and growth simulation of AuCu clusters. Faraday Discuss. **138**, 49–58 (2008)
41. Verlet, L.: Computer «experiments» on classical fluids. I: thermodynamical properties of Lennard-Jones molecules. Phys. Rev. **159**(1), 98–103 (1967)
42. Verlet, L.: Computer «experiments» on classical fluids II Equilibrium correlation functions. Phys. Rev. **165**(1), 201–204 (1968)
43. Wales, D.J., Doye, J.P.K.: Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. J. Phys. Chem. A **101**(28), 5111–5116 (1997)
44. Wu, X., Sun, Y.: Stable structures and potential energy surface of the metallic clusters: Ni, Cu, Ag, Au, Pd, and Pt. J. Nanopart. Res. **19**, 201 (2017)
45. Zaporozhets, T.V., Gusak, A.M., Podolyan, O.M.: Evolution of pores in nanoshells—a competition of direct and inverse Kirkendall effects, Frenkel and Gibbs–Thomson effects: the phenomenological description and computer simulation. Prog. Phys. Metals **13**(1), 1–70 (2012) (in Russian)
46. Zhang, Z., Chen, C., Liu, G., et al.: Enhancement of bonding strength in Ag sinter joining on Au surface finished substrate by increasing Au grain-size. Appl. Surf. Sci. **485**, 468–475 (2019)

An Analysis on the Degrees of Freedom of Binary Representations for Solutions to Discretizable Distance Geometry Problems

Antonio Mucherino

Abstract The degrees of freedom in special binary representations for instances of the Discretizable Distance Geometry Problem (DDGP) are studied in this note article. The focus is on DDGP instances where the underlying graphs, together with their associated vertex orders, are able to satisfy the so-called consecutivity assumption. This additional assumption, in fact, makes it possible to group together subsets of consecutive binary variables, which turn out to strongly depend on each other, so that they can actually be replaced by a smaller subset of binary variables. As a consequence, new binary representations, with reduced degrees of freedom w.r.t the trivial binary representations for DDGP instances, can be introduced and potentially be exploited in DDGP solution methods.

1 Introduction

Given a positive integer K, the Discretizable Distance Geometry Problem (DDGP) [5] asks whether a weighted undirected graph $G = (V, E, d)$ satisfying the following assumptions:

(a) $G[\{1, 2, \ldots, K\}]$ is a clique;
(b) $\forall v \in \{K+1, \ldots, |V|\}$, there exist $u_1, u_2, \ldots, u_K \in V$ such that

- **(b.1)** $u_1 < v, u_2 < v, \ldots, u_K < v$;
- **(b.2)** $\{\{u_1, v\}, \{u_2, v\}, \ldots, \{u_K, v\}\} \subset E$;
- **(b.3)** $\mathcal{V}_S(u_1, u_2, \ldots, u_K) > 0$ (if $K > 1$),

admits a realization $x : v \in V \longrightarrow x_v \in \mathbb{R}^K$ such that, for all $\{u, v\} \in E$, the distance constraints $||x(u) - x(v)|| = d(u, v)$ are satisfied (the symbol $||\cdot||$ indicates the Euclidean norm). The vertices $u_1, u_2, \ldots, u_K$, for every selected vertex $v > K$, are referred to as *reference vertices*, and the corresponding distances $d(u_1, v)$, $d(u_2, v)$, $\ldots$, $d(u_K, v)$ are generally referred to as *reference distances*. Notice that $G[\cdot]$ is

A. Mucherino (✉)
IRISA, University of Rennes 1, Rennes, France
e-mail: antonio.mucherino@irisa.fr

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_13

the subgraph induced by a subset of vertices of V, and $\mathcal{V}_S(\cdot)$ is the volume of the simplex generated by a valid realization of the vertices $u_1, u_2, \ldots, u_K$. Notice that a total *vertex ordering* on the vertices of the graph is implicitly given.

By exploiting the discretization assumptions **(a)** and **(b)**, it is possible to construct a search tree containing all solutions to the DDGP. This work is focused on DDGPs where the distance information can be considered as *exact*, which is: "extremely precise", and consequence of this fact is that the search tree is binary. The DDGP is an NP-hard subclass [1] of the graph embeddability problem, which was proved to belong to the NP-hard class by Saxe in [8].

Over the last years, several variants of the DDGP have been proposed in the scientific literature [2]. The variant which was introduced with the acronym DMDGP[1] is based on an additional assumption on G, where it is required that the reference vertices are the ones that immediately precede the current vertex v in the given vertex ordering. In some previous publications, this additional assumption is named *the consecutivity assumption*. When this assumption is satisfied, it is possible to define a sequence of $(K+1)$-cliques of G which admit an overlap of K vertices [4]. Every clique groups together every vertex $v > K$ with its reference vertices u_1, u_2, $\ldots$, u_K, so that the feasibility of each of them can be verified independently, with an overall computational complexity which is polynomial.

The focus of this contribution is on binary representations for solutions to DDGPs where the consecutivity assumption is satisfied. Actually, it is rather trivial to verify that all instances of the DDGP admit a binary representation, because the paths over the branches of the binary trees that are employed for the representation of the DDGP search domain can be simply identified by vectors of binary variables [6]. In addition to this, however, the consecutivity assumption is exploited in this work to group together subsets of consecutive vertices which are not independent, so that the actual number of degrees of freedom in the binary representations can be reduced. To this aim, a theoretical result previously presented in [3] is exploited. Although the current work may seem a rather light contribution to the research on the DDGP, the use, in any solution methods, of the proposed binary representations can potentially give important benefits for the solution of DDGP instances.

After a short review on solution methods for the DDGP (see Sect. 2), binary representations for DDGP solutions will be studied in Sect. 3. Finally, Sect. 4 will conclude this note article with some future works.

2 Current DDGP Solution Methods

The majority of proposed methods for the DDGP are based on a standard branch-and-prune (BP) algorithmic framework [2]. As mentioned in the Introduction, the discretization assumptions make it possible to represent solutions to DDGP instances

[1] The additional "*M*" stands for *Molecular* and it reminds that this class of problems seemed, when introduced, to be particularly suitable to structural biology problems.

as a discrete (and finite) domain having the structure of a tree. The idea therefore, in all BP frameworks, is to explore such a tree in a depth-first fashion by exploiting the distance information which is ensured by the discretization assumptions; any additional distance information is subsequently used for pruning infeasible tree branches: the distances used to perform this "pruning" action are generally named *pruning distances*. When a leaf node of the search tree is reached, one solution to the instance is given by the path over the tree edges leading from the tree root node to the current leaf node [7].

Several methods based on this algorithmic framework have been proposed over the last years [2]. In none of these methods, however, the pruning distances are exploited to perform a reduction on the degrees of freedom for the binary representation associated to the solutions. When the BP framework is implemented, the depth-first search is supposed to perform its exploration layer by layer, without using possible local information about subgroups of vertices that may have already been explored in previous steps. The reduction on the degrees of freedom is actually possible when this "local information" is exploited (see next section).

3 A Binary Representation for DDGP Solutions

The search domain for a DDGP instance with exact distances is a binary tree [5]. One solution to the problem can be represented as a path from the root to one of its leaf nodes, where layer by layer, the path has only two possible "ways to go": either it may take the left-handed branch, or the right-handed one. A vector of binary variables b_i can therefore be employed to represent one possible solution to a given DDGP instance [7]:

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	b_{14}	…	…	b_n

where b_i corresponds to the vertex v_i, and where the indices i reflect the vertex order associated to the graph G. An immediate observation is that, since the initial clique is generally fixed in solution methods (see assumption **(a)** in the Introduction), the first K binary variables in this binary representation are actually not necessary. Moreover, all DDGP instances admit search trees where a symmetry is present at layer $K + 1$: if the instance is feasible, both binary options 0 and 1 are feasible for the variable b_{K+1}, so that it can actually be neglected (once solutions have been found with $b_{K+1} = 0$, all others can be obtained by changing its value to 1).

The following representation is therefore more convenient (the example is given for $K = 3$):

b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	b_{14}	…	…	b_n

There are $|V| - K - 1$ degrees of freedom in this representation, corresponding to $2^{|V|-K-1}$ potential solutions to be explored by DDGP solution methods. This initial result is not new and is valid for any kind of DDGP instance.

When the DDGP instance satisfies the consecutivity assumption, every subsequence of consecutive $K+1$ vertices corresponds to a $(K+1)$-clique of G, where the first K vertices play the role of reference when constructing the possible positions for the vertex which is the last, in the given vertex ordering, in the clique (see Introduction). The feasibility of all these cliques can be verified in advance: given a binary representation, the only infeasibility that can be detected is related to possible violations on the "pruning distances".

Suppose now that there exists such a pruning distance between the vertex v_6 and v_{12} of the example above (which is, between the binary variables b_6 and b_{12}). Consequence of the consecutivity assumption is the possibility to state that the sub-instance $G[\{v_6, \ldots, v_{12}\}]$ is itself a DDGP instance satisfying the consecutivity assumption. Moreover, this sub-instance has a special property: its first and last vertex (in the vertex ordering inherited from G) are connected by a pruning distances. The symmetry results in [3] indicate therefore that there exist only two (symmetric) solutions to the sub-instance $G[\{v_6, \ldots, v_{12}\}]$. Thus, once this sub-instance is solved independently and its two solutions are encoded by a binary representation, say $c_9(0) = (\hat{b}_9, \hat{b}_{10}, \hat{b}_{11}, \hat{b}_{12})$ and $c_9(1) = \neg c_9(0) = (\neg\hat{b}_9, \neg\hat{b}_{10}, \neg\hat{b}_{11}, \neg\hat{b}_{12})$, the new binary representation for the original instance is:

b_5	b_6	b_7	b_8	c_9	b_{13}	b_{14}	$\ldots$	$\ldots$	b_n

which replaces 4 binary variables with only one. In fact, the binary variables b_6, b_7 and b_8 are not replaced (recall that $K = 3$ in the example), because they represent the given set of values for the initial clique of the sub-instance.

In general, this kind of manipulation on the binary representations makes it possible to perform a reduction on the degrees of freedom by combining subsets of consecutive vertices that depend on each other. For every degree of freedom that is removed from the representation, there is a consequent reduction of one order of magnitude on the number of total DDGP solutions that can be represented. If $m > K+1$ is the number of vertices forming the sub-instance, the degrees of freedom become equal to $|V| - m$ after the first modification, so that 2^{m-K-1} undesired solutions are "erased" from the search tree. Moreover, this operation can be performed for all pruning distances in the DDGP instance, starting from the ones defining smaller sub-instances, up to the larger ones. Even if it remains exponential, the reduction of the corresponding computational complexity can become important, so that any solution method for the DDGP can find a great benefit in using these new binary representations.

It is important to remark that this complexity reduction is possible because every sub-instance delimited by a pruning distance is solved in advance. Naturally, the solution of all these sub-instances introduces an extra computational cost. However, when the overall complexity is computed, one needs to *sum up* all these complexities, for each identified sub-instance, as well as for the final representation of the original instance. This is different from the computation of the complexity associated to the trivial representation consisting of only b_i's variables, where all these single complexities would need to be *multiplied* by each other to obtain the total complexity.

4 Conclusions and Perspectives

This short contribution shows that the properties of DDGP instances satisfying the consecutivity assumption can be exploited to reduce the degrees of freedom in suitable binary representations of its solutions, and in turn decrease the size of the search space that is explored by solution methods. This short article is presented in a simple style by making reference to a running example: future works will be aimed at a formalization of the content of this article, as well as at the possible extension of (at least a part of) these results to DDGPs instances for which the consecutivity assumption is not satisfied. An analysis on DDGP instances where not all its distances can be considered as "exact" will also be investigated.

Acknowledgements This work is partially supported by the international project MULTIBIOSTRUCT funded by the ANR French funding agency (ANR-19-CE45-0019).

References

1. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: The discretizable molecular distance geometry problem. Comput. Optim. Appl. **52**, 115–146 (2012)
2. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. SIAM Rev. **56**(1), 3–69 (2014)
3. Liberti, L., Masson, B., Lee, J., Lavor, C., Mucherino, A.: On the number of realizations of certain Henneberg graphs arising in protein conformation. Discret. Appl. Math. **165**, 213–232 (2014)
4. Mucherino, A.: A Pseudo de Bruijn graph representation for discretization orders for distance geometry. In: Ortuño, F., Rojas, I. (eds.), Proceedings of the 3rd International Work-Conference on Bioinformatics and Biomedical Engineering (IWBBIO15), Granada, Spain. Lecture Notes in Computer Science, Lecture Notes in Bioinformatics series, vol. 9043, pp. 514–523 (2015)
5. Mucherino, A., Lavor, C., Liberti, L.: The discretizable distance geometry problem. Optim. Lett. **6**(8), 1671–1686 (2012)
6. Mucherino, A., Lavor, C., Liberti, L., Talbi, E.-G.: A parallel version of the branch & Prune Algorithm for the molecular distance geometry problem. In: IEEE Conference Proceedings, ACS/IEEE International Conference on Computer Systems and Applications (AICCSA10), Hammamet, Tunisia, pp. 1–6 (2010)
7. Mucherino, A., Liberti, L., Lavor, C.: MD-Jeep: an implementation of a branch & Prune Algorithm for distance geometry problems. In: Fukuda, K., et al. (eds.), Proceedings of the 3rd International Congress on Mathematical Software (ICMS10), Kobe, Japan. Lectures Notes in Computer Science, vol. 6327, pp. 186—197 (2010)
8. Saxe, J.: Embeddability of weighted graphs in k-space is strongly NP-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)

Dynamic Programming for the Synchronization of Energy Production and Consumption Processes

Fatiha Bendali, Eloise Mole Kamga, Jean Mailfert, Alain Quilliot, and Helene Toussaint

Abstract Synchronization of heterogeneous processes is a difficult issue in Scheduling area. Related Mathematical Programming models are difficult to handle, because of the large gaps induced by rational relaxation. We consider here renewable energy production and consumption processes, which have to be performed in situ, and so which must be managed in a synchronized way. We first propose a global dynamic programming algorithm which combine dynamic programming schemes designed in order to deal with both production and consumption processes, and next a pipe-line decomposition which speeds the resolution process while emulating the way those processes may collaborate in a flexible way.

Keywords Scheduling · Dynamic programming · Energy management

1 Introduction

Recent trends in Energy management makes appear that the rise of renewable energy technologies also induces the rise of a new generation of *small energy market players*, made of local producers (factories, farms and even individual householders) while are simultaneously consumers (see [1, 5, 14, 16]). This trend raises concerns for

F. Bendali · E. Mole Kamga · J. Mailfert · A. Quilliot (✉) · H. Toussaint
LIMOS CNRS 6158, Labex IMOBS3, Clermont-Ferrand, France
e-mail: alain.quilliot@isima.fr

F. Bendali
e-mail: bendali@isima.fr

E. Mole Kamga
e-mail: elmole@isima.fr

J. Mailfert
e-mail: jean.mailfert@uca.fr

H. Toussaint
e-mail: helene.toussaint@isima.fr

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_14

the main energy producers which have to deal with more and more decentralized markets, and makes arise a demand from those small players for scheduling tools devoted to the efficient handling of such interacting processes.

In the context of the activities of Labex IMOBS3 in Clermont-Ferrand, France, devoted to *Innovative Mobility Technologies and Services*, we are currently involved into a project about the design and control of a local micro-plant for Hydrogen production, which provides autonomous vehicles in Hydrogen fuel. While most Hydrogen production is usually performed through power costly electrolysis processes, researchers rely here on solar power and photolysis ([5, 8, 12]), which make resulting energy cleaner and cheaper. But the productivity of such a process process deeply depends on the intensity of solar illumination. According to this paradigm, energy production/consumption becomes endogenous, and performed according to a kind of closed loop, which requires both an high level synchronization between production and consumption and an efficient management of uncertainty. Few works address this issue (see [1, 9]). Most of them are related to electric vehicles required to minimize their energy consumption (*Green VRP*, *Pollution-Routing Problem*,...: see [3, 10, 11, 13, 16]), while being subject to temporal/resource constraints, and to the scheduling of industrial processes (see [2, 6, 7, 14, 15]) under time-dependent energy costs and access restrictions.

Our purpose here is to address the synchronization issue, which remains really difficult when it comes to scheduling. This issue also arises in other contexts, like those related to the relocation processes inside vehicle sharing systems, to the routing of drone and truck tours devoted to urban logistics, and to the scheduling of some heterogeneous industrial assembly processes. Dealing with resulting models through Mathematical Programming libraries becomes inefficient because of the large gaps induced by the relaxation of the integrality constraint (*the Big M problem*) and designing *ad hoc Branch and Bound* schemes is also difficult because of the lack of efficient bounding scheme.

Because of the IMOBS3 project, we consider here a fleet of small electric vehicles provided with Hydrogen power cells, involved into local logistic tasks, and, on the other side, a micro-plant in charge of local Hydrogen fuel production. Taken as a whole, the problem also involves forecasting and safety management. Still, we restrict ourselves to the scheduling issue, while focusing on the synchronization issues: that means that we consider only one vehicle,required to perform tasks according to a pre-fixed order, which periodically goes back to the micro-plant in order to refuel. The micro-plant has its own production/storage constraints. If we consider vehicle and micro-plant activities in a separate way, then we may deal with them with simple *ad hoc* dynamic programming algorithms. So what we do first is to design a global *Dynamic Programming Scheme* (DPS), which ties together those two algorithms in an exact way. However, though this DPS allows us to state a PTAS (*Polynomial Time Approximation Scheme*) result, it remains time-costly in practice. So we decompose it into two DPS sub-processes, one related to the vehicle, and the other one to the micro-plant which interact through a kind of *pipe-line* process which emulates the way independent production and consumption players may negotiate in order to reach an agreement about a joint synchronized schedule.

The paper is organized as follows: Sect. 2 describes the **EPC** : *Energy Production/Consumption Problem*. Section 3 describes two DPS respectively designed for the management of the vehicle and of the production and Sect. 4 explain the way those DPS may be merged into a global ***DPS-EPC*** algorithm, together with filtering devices. Section 5 addresses the collaborative issue, and describe the way both production and vehicle DPS may be adapted into two interacting ***DPS-Vehicle*** and ***DPS-Prod*** processes. Section 6 is devoted to numerical experiments, which mainly aim at evaluating the error induced by the decomposition scheme and the gain in terms of running time.

2 The Energy Production/Consumption (EPC) Problem

We consider here some vehicle which has to perform internal logistics tasks, while following a route Γ which starts from some *Depot* node and ends in the same way after going through stations $j = 1, \ldots, M$, according to this order. Start-node *Depot* has label 0 and End-node *Depot* has label $M + 1$. The time required by the vehicle in order to move from j to $j + 1$ is equal to t_j, taking into account the time spent by the vehicle in order to perform local tasks. The vehicle may leave *Depot* at time 0 and should finish its route no later than some threshold time $TMax$.

This vehicle is powered by Hydrogen fuel. The capacity of its tank is denoted by C^{Veh} and we know, for any $j = 0, \ldots, M$, the Hydrogen amount e_j required in order to move from station j to station $j + 1$. The initial Hydrogen load of the vehicle is denoted by E_0, and the vehicle is required to end its trip with at least the same energy load. It comes that the vehicle must periodically refuel. Refueling transactions take place at a *micro-plant*, close to *Depot*: The time required by the vehicle in order to move from station j to the micro-plant (from the micro-plant to j) is denoted by d_j (d_j^*); by the same way, the energy required in order to move from j to the micro-plant (from the micro-plant to j) is denoted by ε_j (ε_j^*). Quantities d_j, d_j^* and t_j, as well as quantities ε_j, ε_j^*, e_j are non null and satisfy the *Triangle Inequality* (Fig. 1).

Figure 2 below displays an example of trip performed by the vehicle along station $Depot = 0, 1, 2, 3, 4, 5, 6 = Depot$, while refueling between station 1 and station 2, and next between station 3 and station 4.

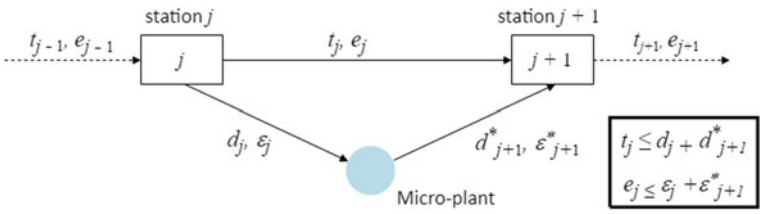


Fig. 1 Time and energy values for 2 consecutive stations j and $j + 1$

On another side, the micro-plant produces Hydrogen *in situ* through photolysis and electrolysis. Resulting fuel is stored inside a tank directly tied to the micro-plant, whose capacity (in energy units) is denoted by C^{MP}. We suppose that the micro-plant is provides with its own time space, which means here that the vehicle time space $\{0, \ldots, TMax\}$ is divided into periods $P_i = [p.i, p.(i+1)[, i = 0, \ldots, N1$, all with a length equal to p such that $TMax = N.p$. We identify index i and period P_i. If the micro-plant is active at some time during period i, then it is active during the whole period i, and produces R_i hydrogen fuel units. At time 0, the load of the micro-plant tank is $H_0 \leq C^{MP}$ and the micro-plant is not active. As for the vehicle, we impose that at time $TMax$, the load of the micro-plant tank will be no less than H_0 and that the micro-plant will not be active.

Because of safety concerns, the vehicle cannot refuel while the micro-plant is producing; Besides, any vehicle refueling transaction should start at the beginning of some period $i = 0, \ldots, N1$, and end at the end of i. Producing Hydrogen fuel has a cost, which may be decomposed into 2 components:

- An *activation cost*, constant and denoted by $Cost^F$, which is charged every time the micro-plant is activated.
- A *time-dependent* production cost $Cost_i^V$ which is independent on the amount of Hydrogen really produced during period i, and reflects the time-indexed prices charged by the electricity provider for the electrolysis process.

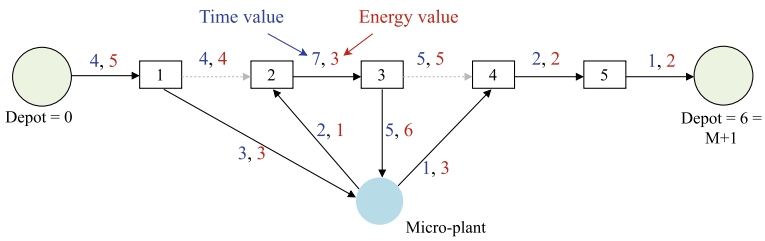


Fig. 2 A vehicle trip, with its refueling transactions

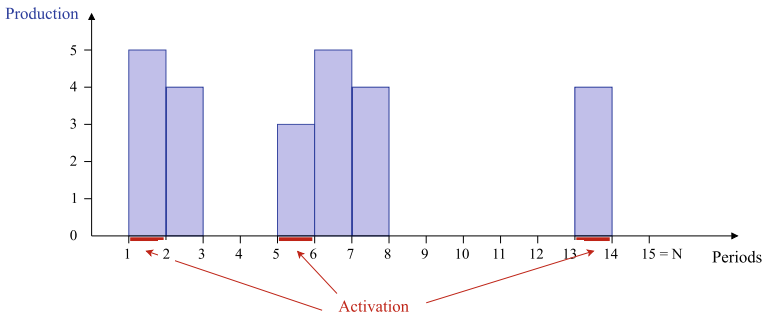


Fig. 3 An example of micro-plant activity, with $N = 15$

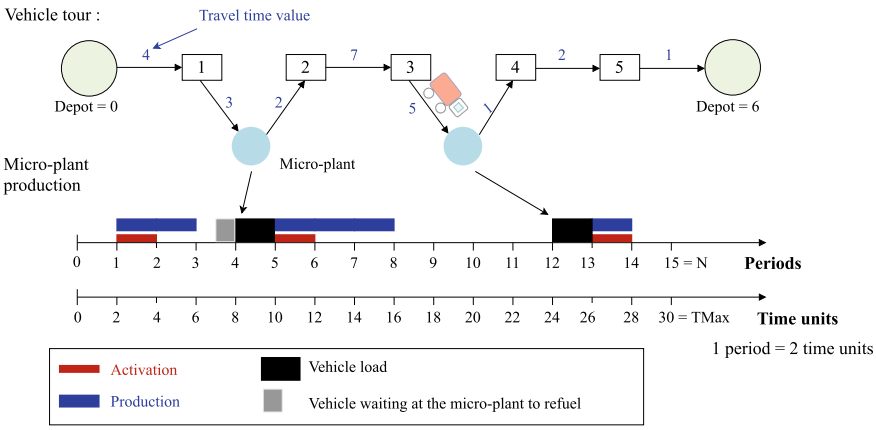


Fig. 4 A feasible solution related to Figs. 2 and 3

Then the *Energy Production/Consumption* (**EPC**) Problem consists in scheduling both the vehicle and the micro-plant in such a way that:

- The vehicle starts from *Depot*$= 0$, visits all stations $j = 1, \ldots, M$ and comes back to *Depot* at some time $T \in [0, TMax]$, while moving to the micro-plant in order to refuel every time it is necessary;
- The micro-plant produces and stores in time the Hydrogen fuel needed by the vehicle;
- Both induced Hydrogen production cost $Cost$ and time T are the smallest possible. We merge both above criteria into a unique one : $Cost + \alpha.T$, where α is some scaling coefficient.

Figure 4 below shows the synchronization between the vehicle and the micro-plant of Figs. 1, 2 and 3, which allows us to get a feasible solution in case $p = 2$, $E_0 = 8$, $H_0 = 4$, $TMax = 30$, $Cost^F = 7$, $C^{MP} = 15$, $C^{Veh} = 15$, $\alpha = 1$.

We easily check that:

Theorem 1 ***EPC*** *is NP-Hard.*

Proof Even when $Cost^F$ is null, **EPC** can be reduced to a Knapsack problem. Left to the reader.

Still, **EPC** lies at the frontier of Time-Polynomiality, as it will appear further.

3 Separately Handling Vehicle and Production Activities

We are going here to show how *Production* and *Vehicle activities* could be independently managed in case while no taking care of any synchronization issue.

3.1 Scheduling the Hydrogen Production Activity

In case no synchronization issue has to be taken into account,*Production* must be scheduled with respect to an estimation V_0 of the demand for the $[0, \ldots, TMax]$ whole period, independently on the time when this demand will be expressed. This can be done through a very simple DPS process :

- The *Time Space* (in the DPS sense) of this DPS process is the period set $I = \{0, \ldots, N\}$. For any such a time value i, a state is defined as a pair (V, Z), with the following meaning:
  - $V =$ load of the micro-plant at the beginning of period i;
  - $Z = 1$ if the micro-plant was active at the end of period $(i - 1)$ and 0 else.
- For any energy amount V, any state value $Z \in \{0, 1\}$ for the micro-plant, and any period number i, we denote by $Cost_Min(i, V, Z)$ the minimum cost required in order to make the micro-plant produce V energy units between time $p.i$ and time $TMax$, Z denoting the state ($Z = 1$ means the micro-plant is active) of the micro-plant at the end of period $i - 1$.
- We may specify the computation of $Cost_Min(i, V, Z)$ through the following backward driven Bellman equations:
  - $Cost_Min(N, V, Z) = 0$ if $V = 0$ and undefined else;
  - For any $i = 0, \ldots, N - 1$, $V \in [0, C^{MP}]$, $Z \in \{0, 1\}$, $Cost_Min(i, V, Z) = Inf[Cost_Min(i{+}1, V, 0), Cost_Min(i{+}1, V{-}R_i, 1) + (Cost^F.(1 - Z) + Cost_i^V]$;
    Related production decision X_Min is equal to 0 (no production during period i) if $Cost_Min(i + 1, V, 0)$ provides us with the value $Cost_Min(i, V, Z)$ and is equal to 1 else.

Those Bellman equations may be implemented through a backward driven DPS in such a way it provides us, for any $i = 0, \ldots, N - 1$, with an accessible set $\Pi(i)$ of relevant values (V, Z), together with related value $Cost_Min(i, V, Z)$ and related decision $z_Min(i, V, Z)$. If $(V, Z) \notin \Pi(i)$, then we extend $Cost_Min$ and z_Min by setting:

- $Cost_Min(i, V, Z) = Cost_Min(i, V^*, Z)$ where V^* is the smallest at least equal to V which is such that $(V, Z) \in \Pi(i)$;
- $z_Min(i, V, Z) = z_Min(i, V^*, Z)$.

3.2 Scheduling the Vehicle Activity: The Vehicle_Driver Problem

So we do not care here about the micro-plant and suppose that it is able to provide at any time the vehicle with as much as energy it needs. Then our goal is to decide the *Refueling Strategy* of the vehicle, that is the $\{0, 1\}$ valued vector $x = (x_j, j =$

$0 \ldots M$) which tells us which stations j are such that the vehicle is going to refuel between j and $j+1$, together with related load vector $L = (L_j, j = 0 \ldots M)$, while minimizing some quantity: $\alpha.T^{End} + \beta.(\sum_j L_j.x_j)$, where ^{End} means the ending date of the vehicle trip, and α, β are non negative coefficients. Coefficient β, which will be discussed later, expresses the part of the fuel cost which will be imputed to the vehicle. We are required to achieve the tour with at least the same initial load E_0 as when we started.

We call the problem the **Simple Vehicle_Driver** (SVD) Problem.

We are first going to prove that **SVD** is time polynomial. In order to do it, we say that a *Refueling Strategy* (x, L) is *Empty Tank* iff:

- Every time but the first time the vehicle refuels, it arrives at the micro-plant with an empty tank;
- It comes back to *Depot* with a load exactly equal to E_0.

Lemma 1 *An optimal Refueling Strategy* (x, L) *may be chosen in such a way it satisfies the Empty Tank Hypothesis*

Proof If (x, L) does not agree with the *Empty Tank* Hypothesis, and makes the vehicle arrive into the *micro-plant* with a non null load $\delta > 0$ in order to perform some current refueling transaction (which is not the first one), then it is possible to make decrease former refueling transaction by δ, and augment current refueling transaction by $\delta \leq \delta$, until cancelling former refueling transaction or making current refueling transaction become *Empty Tank*.

This leads us to define the following oriented graph *VD_Graph*, which express the way the load V of the vehicle evolves when it moves from a station j to a station $j+1$, while complying with the *Empty Tank* hypothesis.

- Nodes of *VD_Graph* are:
  - A source s and a sink $p = ((M+1), E_0)$;
  - Pairs $(0, V)$, $0 \leq V \leq E_0$;
  - Pairs (j, V), $j = 1, \ldots, M$, where V is a non negative number, no larger than C^{Veh};
- Arcs of *VD_Graph* are:
  - Any arc $u = (s, (0, V))$, $V \leq E_0$, with non positive cost $D_u = \beta.(V - E_0)$;
  - Any arc $= ((j, V), (j+1, V - e_j))$, with null cost D_u;
  - Any arc $u = ((j, \varepsilon_j), (j+1, V))$, $V \geq \varepsilon_{j+1}$ with cost $D_u = \alpha.(d_j + d^*_{j+1} - t_j + p) + \beta.(\varepsilon_j + \varepsilon^*_{j+1} - e_j)$.

The meaning of this construction comes through following Lemma 2:

Lemma 2 *Solving* ***SVD*** *means searching for a shortest path from s to p in VD_Graph.*

Table 1 Entrées du *Useful VD_Subgraph*

j	0	1	2	3	4	5=0
e_j	3	4	2	2	3	*
$\varepsilon_j = \varepsilon_j^*$	2	3	2	1	2	2
t_j	4	5	3	2	6	*
$d_j = d_j^*$	3	3	4	2	3	4

Proof We see that:

- Nodes of *VD_Graph* corresponds to the possible states of the vehicle when it performs his trip from *Depot*$= 0$ to *Depot*$= M + 1$, while visiting stations $j = 1, \ldots, M$ and periodically refueling. In case the vehicle has never been refueling before j, $V \leq E_0$ means the hydrogen amount it is going to consume before refueling; Else it means the current hydrogen load of the vehicle at j.
- According to this, we understand the meaning of the arcs:
  - Arc $u = ((j, V), (j + 1, V - e_j))$, with null cost D_u, means that the vehicle move directly from j to $j + 1$;
  - Arc $u = ((j, \varepsilon_j), (j + 1, V))$, $V \geq \varepsilon_{j+1}$ with cost $D_u = \alpha.(d_j + d_{j+1}^* - t_j) + \beta.(\varepsilon_j + \varepsilon_{j+1}^* - e_j)$, means that the vehicle move from j to $j + 1$ while moving first to the micro-plant in order to refuel and arriving to the micro-plant with an empty tank: the cost of such a move is the additional cost (energy + time) induced by this detour;
  - Arc $u = (s, (0, V))$, with $V \leq E_0$, and negative cost $D_u = (V - E_0)$, means the a fictitious decision that the vehicle takes at the beginning of the process in order to make in such a way that it will perform its first refueling transaction with an empty tank. The non positive cost $D_u = \beta.(V - E_0)$ will correspond to the fact that the vehicle will perform this first refueling transaction while arriving to the *micro-plant* with an excess load $(E_0 - V)$.

Then we deduce that any *Refueling Strategy* (x, L) which satisfies the *Empty Tank* Hypothesis gives rise to a path in *VD_Graph*, whose cost is exactly the cost of the Refueling Strategy (x, L), and conversely.

As a matter of fact, we do not need the whole graph *VD_Graph* in order to handle **VD**. If we perform the search for a shortest path from s to p in *VD_Graph* through a backward driven Bellman process, then we see that we only deal with a node collection $X(j)$, $j = 0, \ldots, M + 1$, which may be defined in a recursive way:

- $X(M + 1)$ is reduced to $p = ((M + 1), E_0)$; $X(M) = \{(M, \varepsilon_M), (M, E_0 + e_M)\}$;
- For $j = 0, \ldots, M - 1$, $X(j) = \{(j, \varepsilon_j)\} \cup \{(j, V + e_j)$, for all $(j + 1, V) \in X(j + 1)$ such that $V + e_j \leq C^{Veh}\}$.

If we set $X_Useful = \{s\} \cup (\cup_{j=0,\ldots,M+1} X(j))$, we get the Useful VD-Subgraph as being the subgraph of *VD_Graph* induced by *X_Useful* (Table 1).

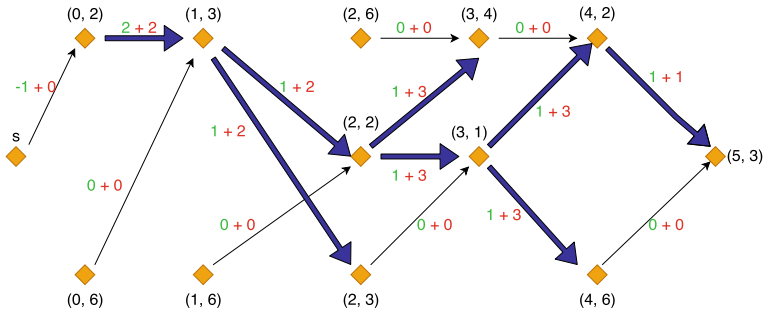


Fig. 5 The *Useful VD_Graph subgraph*

Example 1 The Useful VD_Subgraph in case $M = 4$, $C^{Veh} = 6$, $E_0 = 3$, $\alpha = \beta = 1$, and values t, d, e, ε given by the table below:

Thick arrows represent here the moves of the vehicles which involve a refueling detour. For every arc, $(a + b)$ means the sum of respectively the energy and the time components of the cost of the arc (Fig. 5).

Then *VD_Graph* construction and Lemma 2 allow us to state:

Theorem 2 *SVD can be solved in polynomial time.*

Proof Because of Lemmas 0 and 1 and the above construction of the *Useful VD_Subgraph*, we see that solving **VD** means searching a shortest path from s to p in the *Useful VD_Subgraph*. But, for any j, the cardinality of $X(j)$ does not exceed $M - j + 1$, and the number of arcs which connects $X(j)$ to $X(j + 1)$ does not exceed $Card(X(j) + Card(X(j + 1))$. We conclude.

We implement Lemma 2 through a DPS: *Dynamic Programming Scheme* **DP_BVD**, which scans the time space $J = \{0, 1, \ldots, M, M + 1\}$ in a backward way, according to the following Bellman: equations:

- $X(M + 1) = \{E_0\}$; $W^{Veh}_{\alpha,\beta}(M + 1, E_0) = 0$;
- For any $j = M, \ldots, 0$, $V^{Veh} \in X(j)$ if $V^{Veh} - e_j \in X(j + 1)$ Or $V^{Veh} = \varepsilon_j$;
- For any $j = M, \ldots, 0$, and any $V^{Veh} \in X(j)$:
  - If $V^{Veh} \neq \varepsilon_j$ then $W^{Veh}_{\alpha,\beta}(j, V^{Veh}) = W^{Veh}_{\alpha,\beta}(j + 1, V^{Veh} - e_j) + \alpha t_j$;
  - If $V^{Veh} = \varepsilon_j$ then:
    - Set $C_1 = Inf_{V \in X(j+1)}(W^{Veh}_{\alpha,\beta}(j + 1, V) + \alpha.(d_j + p + d^*_{j+1}) + \beta.(V + \varepsilon^*_{j+1}))$;
    - Set $C_2 = W^{Veh}_{\alpha,\beta}(j + 1, V^{Veh} - e_j) + \alpha t_j$ if $V^{Veh} - e_j \in X(j)$ and $C_2 = +\infty$ else;
    - $W^{Veh}_{\alpha,\beta}(j, V^{Veh}) = Inf(C_1, C_2)$.

Then we see that the optimal value of **SVD** is equal to: $Inf_{V \in X(0), V \leq E_0}(W^{Veh}_{\alpha,\beta}(0, V) - \beta.(E_0 - V))$, while related refueling strategy (x, L) derives from above Bellman equations in a standard way. Also, we may extend $W^{Veh}_{\alpha,\beta}$ to any pair j, V^{Veh} by setting: $W^{Veh}_{\alpha,\beta}(j, V^{Veh}) = Inf_{V \in X(j+1), V \leq V^{Veh}}(W^{Veh}_{\alpha,\beta}(j, V) - \beta.(V^{Veh} - V))$. Then we see that for any $j = 0, \ldots, M$ and any value V^{Veh}:

- $W^{Veh}_{1,0}(j, V^{Veh})$ provides us with a lower bound for the time required from the vehicle in order to achieve its trip when it starts moving from station j with a tank load equal to V^{Veh};
- $W^{Veh}_{1,0}(j, V^{Veh})$ provides us with a lower bound for the energy amount the vehicle will have to refuel in order to achieve its trip when it starts moving from station j with a tank load equal to V^{Veh}.

4 Linking Production and Vehicle DPS into a Unique Global Dynamic Programming Scheme

The purpose of this section is to show the way both Dynamic Programming scheme designed in order to separately deal with the scheduling of production and with the **BVD** problem may be synchronized in order to give rise to a unique algorithm **DP_EPC**, which will globally handle our **EPC** problem in an exact way. Once again, we are going to describe **DP_EPC** by providing its main DPS components (Time Space, State Space, etc.) and explaining the way they derive from Sect. 3 through some composition mechanisms.

DP_EPC Time Space: It is the product $\Pi = I.J$ of the time spaces I and J which we used in order to deal with respectively production scheduling and the **BVD** Problem. So a time pair is a pair (i, j), $i = 0, \ldots, N$, $j = 0, \ldots, M + 1$. Π is provided with its standard partial ordering.

DP_EPC State Space: For any such a time pair (i, j), related state space is the product of the state spaces respectivey involved into production scheduling and vehicle handling. That is that a state at time (i, j) is a 4-uple $s = (Z, T, V^{Tank}, V^{Veh})$, with:

- $Z = 1$ means the micro-plant is active at the end of period $i - 1$, that means at time $p.i$.
- V^{Tank} and V^{Veh} are respectively the loads of the micro-plant tank at the beginning of period i and the load of the vehicle tank when it arrives at station j;
- T is a value in $0, \ldots, TMax$.

Synchronization is achieved through following relations $<<$, $>>$ and $==$, which link real time values T with DPS time pairs (i, j):

- $T << i$ if $T < p.i$: It means that the vehicle is between j and the micro-plant, possibly waiting for being refueled;

- $T >> i$ if $T \geq p.(i+1)$: It means that then the vehicle is on the road to j, which it shall reach at time T;
- $T == i$ if $p.i \leq T < p.(i+1)$: It means that the vehicle is located in j, and decides between keeping on to $j+1$ or moving to the micro-plant.

Decisions: Then a decision D at time pair (i, j) and applied to a state $s = (Z, T, V^{Tank}, V^{Veh})$ is a 3-uple $D = (z, x, \delta)$ in $\{0, 1\}^3$, with the meaning:

- $z = 1$ means as in Sect. 3.1, it means that the micro-plant decides to produce during period i;
- x refers to a decision taken only when $T == i$: in such a case, $x = 0$ means as in that the vehicle moves from station j to station $j+1$ without refueling, while $x = 1$ means that it refuels at the micro-plant while traveling from j to $j+1$;
- δ means the synchronization component of decision vector D: $\delta = 1$ means the vehicle is located at the micro-plant and decides to refuel during period i, forbidding the micro-plant to be active during this period. It requires $T << i$ and $p.i - T \geq d_j$.
- Notice no load component L is involved here: it is due to the fact that we are going to perform our DPS according to a forward driven strategy and that, in case $\delta = 1$, the Empty Tank principle of Sect. 3.2 will dictate us the optimal value of related load L.

Transitions : Decision D is taken at the end of period $i-1$. We see that, for any time pair (i, j) and state $s = (Z, T, V^{Tank}, V^{Veh})$, no more than 4 decisions may be feasible:

- **1 th case**: $T >> i$. Then the only choice is about z. Setting $z = 1$ requires $V^{Tank} + R_i \leq C^{MP}$. We shift from (i, j) to $(i+1, j)$, resulting state is $s' = (1, T, V^{Tank} + z.R_i, V^{Veh})$ and transition cost is $z.(Cost^F.(1-Z) + Cost_i^V.)$.
- **2 th case**: $T << i$ and $p.i - T < d_j$. Then the vehicle is moving from j to the micro-plant, and it cannot refuel yet. Once again, the only choice is about z, with the same preconditions and costs as in the first case.
- **3 th case**: $T << i$ and $p.i - T \geq d_j$. Then, we have 3 possibilities:
  - **Producing**: $z = 1$; $\delta = 0$. It requires $V^{Tank} + R_i \leq C^{MP}$ and $p.(i+1) + p + d^*_{j+1} + \sum_{k \geq j+1} t_k \leq TMax$. Resulting states and costs are in the first case.
  - **Refueling**: $z = 0$; $\delta = 1$. The empty tank principle means that the vehicle refuels an energy amount L as large as possible, but in the case when it can come back to $M + 1 = Depot$ without refueling anymore:
    In case $E_0 + \varepsilon^* j + 1 + \sum_{k \geq j+1} e_k \leq Inf(C^{Veh} - V^{Veh}, V^{Tank})$ then $L = E_0 + \varepsilon^* j + 1 + \sum_{k \geq j+1} e_k - V^{Veh})$ Else $L = Inf(C^{Veh} - V^{Veh}, V^{Tank})$.
    It requires $\varepsilon_{j+1} + \varepsilon^* j + 1 \leq L$. We shift from (i, j) to $(i+1, j+1)$ and resulting state is s'=$(0, p.(i+1) + d^*_{j+1}, V^{Tank} - L, Inf(C^{Veh} - \varepsilon^* j + 1, V^{Veh} + L - \varepsilon^* j + 1))$. Transition cost is $\alpha.(p.(i+1) + d^*_{j+1} - T)$.
  - **Doing nothing**: $z = 0$; $\delta = 0$. It requires $p.(i+1) + p + d^*_{j+1} + \sum_{k \geq j+1} t_k \leq TMax$. Then we shift from (i, j) to $(i+1, j)$, resulting state is s'=$(0, T, V^{Tank}, V^{Veh})$, and transition cost is null.

- **4 th case**: $T == i$. Then we have 4 possibilities:
  - **Producing and keeping the vehicle towards** $j+1$: $z=1$, $x=0$. It requires $V^{Tank}+R_i \leq C^{MP}$, $V^{Veh}-e_j-\varepsilon_{j+1} \geq 0$ and $T+t_j >> i$. Then we shift from (i, j) to $(i+1, j+1)$, resulting state is $s' = (0, T+t_j, V^{Tank}R_i, V^{Veh}-e_j)$ and transition cost is $z.(Cost^F.(1-Z)+Cost_i^V.)+\alpha.t_j$.
  - **Not Producing and keeping the vehicle towards** $j+1$: $z=1$, $x=0$. It requires $V^{Veh}-e_j-\varepsilon_{j+1} \geq 0$. Then, if $T+t_j >> i$, we shift from (i, j) to $(i+1, j+1)$, resulting state is $s' = (0, T+t_j, V^{Tank}V^{Veh}-e_j)$ and transition cost is $\alpha.t_j$. Conversely, if $T+t_j == i$, then we shift from (i, j) to $(i, j+1)$, which means that i does not evolve, in such a way we may decide to move to the micro-plant once it arrives in $j+1$. Resulting state and transition cost are the same as in the case $T+t_j >> i$.
  - **Not Producing and moving the vehicle to the micro-plant**: $z=0$, $x=1$. It requires $Sup(p.(i+1), T+d_j)+d^*_{j+1}+p+\sum_{k\geq j+1} t_k \leq TMax$. Then we shift from (i, j) to $(i+1, j)$, resulting state is $s' = (0, T, V^{Tank}, V^{Veh}-\varepsilon_j)$and transition cost is null.
  - **Producing and moving the vehicle to the micro-plant**: $z=1$, $x=1$. It requires $V^{Tank}+R_i \leq C^{MP}$ and $Sup(p.(i+1), T+d_j)+d^*_{j+1}+p+\sum_{k\geq j+1} t_k \leq TMax$. Then we shift from (i, j) to $(i+1, j)$ resulting state is $s' = (1, T, V^{Tank}+R_i, V^{Veh}-\varepsilon_j)$ and transition cost is $(Cost^F.(1-Z)+Cost_i^V)$.

Initial and final states: Initial state corresponds to time pair (0, 0) and 4-uple $s_0 = (0, 0, H_0, E_0)$. Final state corresponds to any time pair $(i \leq N, M+1)$, and any 4-uple $(Z, T \leq TMax, V^{Tank} \geq H_0, V^{Veh} \geq E_0)$.

Table 2 below provides us with the evolution of time pairs (i, j), states $s = (Z, T, V^{Tank}, V^{Veh})$, decisions $D = (z, x, \delta)$ and costs $Cost+\alpha.T$, induced by the solution described in Fig. 4.

Bellman Equations: With every time pair (i, j) and any state $s = (Z, T, V^{Tank}, V^{Veh})$, we associate its Bellman value W (smallest cost of a sequence of transitions from initial state s_0 at time (0, 0) to state s at time (i, j)). Then we implement our DP_EPC algorithm according to a **forward driven** strategy. For any current time pair (i, j) we denote by S(i, j) the active state subset related to (i, j) and which is in fact a set of 3-uples $(s, W, Decision)$ where W is the above defined value and Decision means the decision (z, x, δ) which gave rise to s.

Then, (i, j) being the current time pair, we scan S(i, j), and for any such a state $s = (Z, T, V^{Tank}, V^{Veh})$, we do:

1. Generate the related feasible decision set Dec((i, j), s) of all decision $D = (z, x, \delta)$ which may be applied to state s at time pair (i, j);
2. For every $D = (z, x, \delta)$ in $Dec((i, j), s)$ do
   Generate resulting time pair (i, j') and state $s' = (Z', T', V^{Tank\prime}, V^{Veh\prime})$, together with the value $W+CT(D)$ where $CT(D)$ means the cost of the transition induced by D;

Table 2 Simulation of the sequence of states and decisions related to Fig. 4

Time pair (i,j)		State $s = (Z, T, V^{Tank}, V^{Veh})$				Solution cost W	Decision $D = (z, x, \delta)$		
i	j	Z	T	V^{Tank}	V^{Veh}	W=Cost + T	z	x	δ
0	0	0	0	4	8	0 + 0	0	0	0
1	1	0	4	4	3	0 + 4	1	0	0
2	1	1	4	9	3	8 + 4	1	1	0
3	1	1	4	13	3	9 + 4	0	0	0
4	1	0	4	13	3	9 + 4	0	0	1
5	2	0	12	0	12	9 + 12	1	0	0
6	3	1	19	3	9	18 + 19	1	0	0
7	3	1	19	8	9	20 + 19	1	0	0
8	3	1	19	12	9	22 + 19	0	0	0
9	3	0	19	12	9	22 + 19	0	1	0
10	3	0	19	12	9	22 + 19	0	0	0
11	3	0	19	12	9	22 + 19	0	0	0
12	3	0	19	12	9	22 + 19	0	0	1
13	4	0	27	0	12	22 + 27	1	0	0
14	5	1	29	4	10	30 + 29	0	0	0
15	6	0	30	4	8	30 + 30	*	*	*

- If state s' is not involved in current set $S(i', j')$ then insert it into $S(i', j')$, together with value $W + CT(D)$ and decision D;
- If state s' is already involved in $S(i', j')$ with a value $W* > W + CT(D)$, then $W + CT$ becomes the value associated with (i', j') and s' and D becomes related decision;
- If state s' is already involved in $S(i', j')$ with a value $W* \leq W + CT$, then we discard s'.

4.1 Logical Filtering Devices

As usually when it comes to Dynamic Programming, we may have to deal with very large numbers of states, as soon as N and M increase, and so we need to introduce filtering or pruning devices. Clearly, the following strong dominance rule may be applied:

Strong Dominance Rule: For a given time pair (i, j), if 2 related states $s_1 = (Z_1, T_1, V_1^{Tank}, V_1^{Veh})$ and $s_2 = (Z_2, T_2, V_2^{Tank}, V_2^{Veh})$ given together with values W_1 and W_2 are such that:

$W_1 \leq W_2$; $T_1 \leq T2$; $Z_1 \geq Z_2$; $V_1^{Tank} \geq V_2^{Tank}$; $V_1^{Veh} \geq V_2^{Veh}$;

then s_1 strongly dominates s_2, and we kill s_2 (i.e. we remove it from the list of the states associated with (i, j)).

This Strong Dominance rule has little filtering power, since it is too restrictive. Still, we may derive from Sect. 3 more powerful filtering devices; In order to do it, we pre-compute and store the values $W_{1,0}^{Veh}(j, V^{Veh})$, and $W_{0,1}^{Veh}(j, V^{Veh})$ of Sect. 3.2, together with values $Prod_Max(i) = \sum_{i1 \geq i} R_i 1$. Then, given some a time pair (i, j), and some state $s = (Z, T, V^{Tank}, V^{Veh})$ at the beginning of period i, we may apply the following (logical) filtering rules:

- **Makespan Based filtering rule**: If $(T + W_{1,0}^{Veh}(j, V^{Veh}) \geq (TMax + 1)$ then we may kill state $s = (Z, T, V^{Tank}, V^{Veh})$ related to time pair (i, j), since there will not be enough time left for the vehicle to achieve its trip before deadline $TMax$.
- **Energy Based filtering rule**: If $W_{0,1}^{Veh}(j, V^{Veh}) > Prod - Max(i) + V^{Tank}$ then kill state $s = (Z, T, V^{Tank}, V^{Veh})$ related to time pair (i, j), since it will not be possible to produce the energy that the vehicle will need in order to achieve its trip.

4.2 *Quality Based Filtering Devices: A Greedy Version of DP_EPC*

Another way to filter is to decide a maximal number Max_State of states for every time pair (i, j), and then to select states in active state subset $S(i, j)$ are those Max_State states which are feasible according to above Makespan and Energy Based rules, and which are the best according to some quality criterion. In order to get such a criterion, we use once again pre-computed values $W_{0,1}^{Veh}(j, V^{Veh})$ and $W_{0,1}^{Veh}(j, V^{Veh})$, together with values $Cost - Min(i, V, Z)$, which we also pre-compute and store, prior to the execution of our **EPC** main algorithms. Let us recall that, for any energy amount V and any period number i, $Cost_Min(i, V, Z)$ provides us with a lower bound for the economic cost induced by the production of V units of energy, starting at the beginning of period i with a micro-plant in state Z.

Given now a time pair (i, j) and some related state $s = (Z, T, V^{Tank}, V^{Veh})$ with value W. Then we get a lower bound LB of the best cost of an EPC trajectory involving time pair (i, j) and state s, by setting:

$LB((i, j), s) = W + \alpha.W_{0,1}^{Veh}(j, V^{Veh}) + Cost - Min(i, W_{0,1}^{Veh}(j, V^{Veh}), Z)$.

Then we may consider that this lower bound provides with an ad hoc quality criterion. This leads us to turn our DP_EPC algorithm into a greedy **GREEDY_EPC** Algorithm 1.

In instruction 13 of Algorithm 1, s' is not dominated in the sense of the Strong Dominance rule by any s_1 in $S(i', j')$ and the number of existing states s_1 in $S(i', j')$ which are such that $LB((i', j'), s') \geq LB((i', j'), s_1)$ is smaller than Max_State.

If we come back now to our main DP_EPC algorithm, then we see that GREEDY-EPC(Max_State) provides us with an EPC upper bound Greedy_Value, which we may use in order to implement the following Quality Based filtering rule.

Algorithm 1 GREEDY-EPC(Max_State)

```
Initialize time value (i, j) ← (0, 0) and active state subset S(0, 0) = ((0, 0, H_0, E_0), W_0 = 0);

Initialize other active state subset S(i, j) to Nil, Not Fail;
while (i, j) ≠ (N, M + 1) ∧ not Fail do
  if S(i, j) = Nil then
    Fail ;
  else
    for any s ∈ S(i, j) do
      Compute feasible decision set Dec((i, j), s);
      for any D = (z, x, δ) ∈ Dec((i, j), s) do
        Generate resulting time pair (i, j′), state s′ = (Z′, T′, V^Tank′, V^Veh′), and value W′;

        Apply both the Makespan Based and Energy Based filtering rules;
        Compute LB((i′, j′), s′);
        If s′ is not dominated then Insert s′ into S(i′, j′);
        (i, j) ← Successor(i, j), in the sense of the canonical ordering on the production
        I.J;
      end for
    end for
  end if
end while
```

Quality Based filtering rule: Given some time pair (i, j), and some state $s = (Z, T, V^{Tank}, V^{Veh})$ at the beginning of period i, If $LB((i, j), s) \geq Greedy_Value$, then kill state $s = (Z, T, V^{Tank}, V^{Veh})$, related to time pair (i, j).

5 Linking Production and Vehicle DPS in a Pipe-Line Collaborative Scheme

Filtering devices of Sects. 4.1 and 4.2 do not keep the number of states involved into the execution of **DP-EPC** to become large as N and M increase. More, scheduling both vehicle and production activities according to a fully centralized paradigm may be criticized for some lack of flexibility and for the fact that it does not fit with the way decision are usually taken in a context, which involves collaborative features.

So what we are going to do now is to describe the way *Production Planner* and *Vehicle Driver* models of Sect. 3 may be linked together in a more flexible way than in Sect. 4, in order to yield a heuristic collaborative handling of the **EPC** Problem, which might be used in case production planner and vehicle driver are independent players.

Let us recall that **DP_BVD** allows to retrieve what we call the *Primary Refueling Strategy*, that means a $\{0 \ldots M\}$ indexed vector $LOAD$, together with a number S, with the following meaning:

- S is the number of refueling transactions performed by the vehicle;
- For $s = 1 \ldots S$, $LOAD[s]$ provides us with a 5-uple $(St, L, TInf, TSup, Lag)$ such that:
  - $St[s]$ is the station j such that the sth refueling transaction is performed between j and $j + 1$;
  - $L[s]$ is the quantity of fuel which is loaded during the loading transaction s;
  - $TInf[s]$ is the earliest time when it may start;
  - $TSup[s]$ is the latest possible date it may start;
  - Lag is the minimal delay between the date when the sth refueling transaction may start and the date when the next one $(s + 1)$ may start (or the end of the trip if $s = S$).

In order to synchronize this *Primary Refueling Strategy* with the Production process, we translate values $TInf$, $TSup$ and Δ in terms of periods $i = 0 \ldots N - 1$, and derive from vector $LOAD$, what we call a *Reduced Refueling Strategy* (m, M, B, μ):

- Lower bounds $m_1, \ldots, m_S$ and upper bounds $M_1, \ldots, M_S$ for the period numbers $i_1, \ldots, i_S \in \{0, \ldots, N - 1\}$ when the refueling transactions take place;
- Time lag coefficients $B_1, \ldots, B_Q$ which reflects the following constraints that those period numbers have to satisfy : for any $s = 1 \ldots S - 1$, $i_{s+1} \geq i_s + B_s$;
- Loads μ_s = quantities of H^2 which is loaded for every value $s = 1 \ldots S$.

We are going to show how this Reduced Refueling Strategy, output of the DP_BVD process, may be used as an input for an Extended_Production process, whose purpose is to adapt the scheduling of the micro-plant activity in order to integrate the vehicle driver preferences.

5.1 The Ext_Prod Extended Production Model

We start from the same hypothesizes as in the Production Model of Sect. 3.1, while supposing that we are provided with a Reduced Refueling Strategy (m, M, B, μ)as defined above, and with some coefficient λ Then our goal becomes to schedule the activity of the micro-plant, that is computing $\{0, 1\}$-valued vector $z = (z_i, i = -1, 0, \ldots, N - 1)$ with the meaning: $z_i = 1$ iff the micro-plant is active during period i, in such a way that:

- The vehicle may refuel S times, quantities $\mu_1, \ldots, \mu_s$ at some periods $i_1, \ldots, i_S$ in a way consistent with time lags $B_1, \ldots, B_Q$ and time window constraints induced by coefficients $m_1, \ldots, m_S$, $M_1, \ldots, M_S$ of the Reduced Refueling Strategy;
- The micro-plant ends loaded with a load at least equal to the quantity H_0 it started with;
- The quantity $\lambda.i_S + \sum_{i=0\ldots N-1}(Cost^F.(z_i.(1 - z_{i-1}) + Cost_i^V \ldots z_i)$ is the smallest possible.

We call this problem the Extended_Production Problem, and denote resulting model by **Ext_Prod**. Actually, main variables inside this model are production vector z and refueling $\{0, 1\}$ vector $\delta = (\delta_i, i = 0, \ldots, N-1)$, whose meaning comes as follows: $\delta_i = 1$ iff the vehicle refuels during period i. Vector δ will ensure the synchronization between the production and the activity of the vehicle such as it is summarized by the Reduced refueling Strategy (m, M, B, μ).

5.2 The DP_Ext_Prod Algorithm

As it was the case for the global **EPC** model and for the **BVD** model, we deal with **Ext_Prod** through a dynamic programming algorithm **DP_Ext_Prod**, whose main components (*Time Space*, *States*, *Decisions*, etc.) come as follows:

DP_Ext_Prod Dynamic Programming Algorithm: Architecture

- **Time Space**: The Time (in the DPS sense) Space is the set $I = \{0 \ldots N\}$.
- **State Space**: For any $i = 0 \ldots N$, a state is defined as a 4-uple $E = (Z, V^{Tank}, Rank, Delay)$, with Rank in $0 \ldots S$:
  - $Z = 1$ means that the micro-plant is active at the beginning of period i (at the end of period $i-1$).
  - V^{Tank} means the current load of the micro-plant tank at the beginning of period i.
  - $Rank \in 0 \ldots S$ means that the refueling transaction with number $Rank$ has been performed and that we are waiting in order to perform next refueling transaction with number $Rank + 1$. It comes that $Rank = 0$ corresponds to a fictitious refueling transaction.
  - $Delay$ means the difference between i and the period at which the Rankth refueling transaction has been performed.

  For every $i = 0 \ldots N$, a state E is provided with its current Bellman value W^{Prod}.

  - **Initial state** is $E^{Start} = (0, H_0, 0, 1)$, with related value $W^{Prod} = 0$, and time value $i = 0$;
  - **Final states** are states $E^{End} = (Z, V^{Tank} \geq H_0, S, p)$, associated with a time value $i \leq N$: notice that the process may not be finished when the last refueling transaction takes place, since the micro-plant may have to keep on producing in order to reach its initial load level H_0.
- **Decisions** : For any $i = 0 \ldots N$, $E = (Z, V^{Tank}, Rank, Delay)$, a decision is defined as a 2-uple $(z, \delta) \in \{0, 1\}^2$, with the following meaning:
  - $z = 1$ means that the micro-plant will produce during period i;
  - $\delta = 1$ means that the vehicle will perform its $(Rank + 1)^t h$ refueling transaction during period i.

Since production ad refueling cannot be performed simultaneously, we see that there are only 3 possible decisions.

- $z = 0, \delta = 0$: Then a precondition is that $(i \leq M_{Rank+1} - 1)$;
- $z = 1, \delta = 0$: Then a precondition is that $(i \leq M_{Rank+1} - 1) \wedge (V^{Tank} + R_i \leq C^{Tank})$;
- $z = 0, \delta = 1$: Then a precondition is that $(V^{Tank} \geq \mu_{Rank+1}) \wedge (M_{Rank+1} \geq i \geq m_{Rank+1}) \wedge (Delay \geq B_{Rank})$.

- **Transitions**: In case decision (z, δ) is feasible, it induces the following transitions:
  - $z = 1,\ \delta = 0$: At time $(i + 1)$ resulting state E_1 will be $E_1 = (1, V^{Tank} + R_i, Rank, Delay + 1)$; If $Rank \leq S - 1$ then related transition cost will be equal to $\lambda + (Cost^F.(1 - Z) + Cost_i^V \ldots z)$, else it will be only equal to $(Cost^F.(1 - Z) + Cost_i^V.z)$;
  - $z = 0, \delta = 0$: At time $(i + 1)$ resulting state E_1 will be $E_1 = (0, V^{Tank}, Rank, Delay + 1)$; If $Rank \leq S - 1$ then related transition cost will be equal to λ else it will be null;
  - $z = 0,\ \delta = 1$: At time $(i + 1)$ resulting state E_1 will be $E_1 = (0, V^{Tank} - \mu_{Rank+1}, Rank + 1, 1)$, and related transition cost will be equal to λ.

- **Search Strategy**: As in the case of DP_EPC, we shall perform a forward driven search, in order ot take advantage of the filtering devices related to functions $W_1^{Veh}, 0(j, V^{Veh})$, $W_0^{Veh}, 1(j, V^{Veh})$ and $Cost_Min(i, V, Z)$.

5.3 The Pipeline Scheme

In order to conclude this Section V, we need now to make both DP_BVD and DP_Ext_Prod processes interact. We could propose several way to do it. The simplest one consists in considering our Vehicle/Production decomposition scheme as a hierarchical decomposition scheme with DP_BVD playing the role of the Master process and DP_Ext_Prod playing the role of the slave. This give rise to the following heuristic pipe-line process:

EPC_PipeLine Algorithm:

Input: All data related to the **EPC** model, including scaling coefficient α.

Output: Production vector $z = (z_i, i = 0 \ldots N - 1)$, Synchronization vector $\delta = (\delta_i, i = 0 \ldots N - 1)$, Refueling vector $x = (x_j, j = 0, \ldots, M)$, together with related **EPC** global cost W.

1. Choose value β;
2. Apply the DP_BVD algorithm to resulting BVD model $\rightarrow$ Get a Primary Refueling Strategy LOAD together with vector x;
3. Derive a Reduced Refueling Strategy (m, M, B, μ);
4. Set $\lambda = \alpha.p$;

5. Apply the DP_Ext_Prod algorithm → Get Production vector z and Synchronization vector δ, together with DP_Ext_Prod value W^{Prod};
6. Derive VAL from W^{Prod} .

Discussion: Choosing β?

Coefficient β should reflect what will be the true production cost par energy unit of the energy amount $\sum_s \mu_s$ that the micro-plant will have to produce once the DP_BVD algorithm have been applied. The problem is that we do not know a priori the distribution in the time space of related production process. So we proceed in an empirical way, while using the information contained in the Cost_Min table and doing as if an optimistic estimation H of $\sum_s \mu_s$ were to be produced in an optimal way, without any temporal restrictions. So we set:

- $H = W_{0,1}^{Veh}(0, E_0)$;
- $Rough - Cost = Cost - Min(0, H, 0)$;
- $\beta = (Rough - Cost/H)$.

This formula means a kind of statistic estimation of the per-unit energy cost one has to accept in order to make possible for the vehicle to achieve its tour.

6 Numerical Experiments

Purpose: The main purpose of those numerical experiments is to provide us with an evaluation of the collaborative EPC_Pipeline algorithm of Sect. 5.3. In order of get this evaluation we compare both values and computational costs obtained through this algorithm with those induced by exact centralized DP_EPC algorithm. Besides, since the main challenge induced by the design of exact DP_EPC algorithm is about the control of computational costs, we test the efficiency of the greedy algorithm centralized GREEDY_EPC.

Technical Context: Algorithms were implemented in C++, on a computer running Windows 10 Operating system with an IntelCore i5-6500@3.20 GHz CPU, 16 Go RAM and Visual Studio 2017 compiler.

Instances: We fix N, M and p, and randomly generate stations j and Depot and the Micro-Plant as point of the R^2 space. Then d_j, d_j^* and t_j, e_j, ε_j, ε_j^* respectively corresponds to Euclidean distance and Manhattan distance roundings, in such a way they take integral values and keep on satisfying the Triangle Inequality. Then we fix C^{MP}, C^{Veh}, in such a way it ensures the existence of a feasible solution. Finally, we fix the cost coefficients, in such a way that the fixed cost $Cost^F$ is at least equal to the largest coefficient $Cost_i^V$, $i = 0, \ldots, N-1$. In what follows, every instance will be represented by an identifier Id and 4_uple (N, M, p).

Table 3 Values Opt versus V_Pipe

Instance Id: (N, M, p)	Opt	V_Pipe	GAP_Pipe
1 : (15, 4, 4)	46	47	2,17
2 : (78, 10, 1)	94	97	3,19
3 : (94, 10, 1)	129	129	0
4 : (114, 10, 1)	131	140	6,87
5 : (99, 10, 1)	157	157	0
6 : (59, 10, 2)	109	113	3,66
7 : (36, 10, 2)	97	103	6,185
8 : (78, 10, 2)	140	140	0
9 : (57, 10, 2)	141	150	6,38
10 : (26, 8, 4)	116	116	0
11 : (26, 10, 4)	133	133	0
12 : (24, 10, 4)	184	187	1,63
13 : (30, 10, 4)	232	243	4,74
14 : (33, 10, 4)	182	182	0
15 : (20, 8, 4)	81	83	2,46
16 : (25, 8, 4)	108	−1	Failure
17 : (27, 8, 4)	100	104	4,00
18 : (50, 10, 4)	131	133	1,52
19 : (24, 10, 4)	102	102	0
20 : (44, 10, 4)	126	126	0
21 : (32, 12, 4)	297	304	2,35
22 : (32, 12, 4)	297	304	2,35
23 : (45, 14, 4)	305	325	6,55
24 : (30, 8, 4)	199	199	0
25 : (26, 8, 4)	141	141	0
26 : (17, 10, 4)	65	65	0
27 : (19, 10, 4)	107	107	0
28 : (20, 10, 4)	121	124	2,47
29 : (50, 12, 4)	202	209	3,46
30 : (50, 12, 4)	202	209	2,17

Outputs: For any instance, we provide:

- Table 3:
  - The optimal EPC value Opt, computed by DP_EPC;
  - The value V_Pipe, computed by EPC_Pipeline, together with related gap GAP_Pipe = (V_Pipe - Opt)/Opt; Value -1 means that the algorithm could not find any feasible solution.

Table 4 Values Opt versus V_Greed(x)

Instance Id: (N, M, p)	Opt	V_Greed(1)	V_Greed(50)	V_Greed(100)	GAP_100
1 : (15, 4, 4)	46	−1	46	46	0
2 : (78, 10, 1)	94	−1	127	103	9,57
3 : (94, 10, 1)	129	−1	173	141	9,30
4 : (114, 10, 1)	131	−1	150	144	9,92
5 : (99, 10, 1)	157	−1	165	163	3,82
6 : (59, 10, 2)	109	−1	117	113	3,66
7 : (36, 10, 2)	97	173	133	97	0
8 : (78, 10, 2)	140	−1	141	176	25,7
9 : (57, 10, 2)	141	−1	154	150	6,38
10 : (26, 8, 4)	116	−1	116	116	0
11 : (26, 10, 4)	133	−1	171	133	0
12 : (24, 10, 4)	184	−1	188	−1	Failure
13 : (30, 10, 4)	232	−1	248	248	6,89
14 : (33, 10, 4)	182	−1	207	201	10,4
15 : (20, 8, 4)	81	−1	87	81	0
16 : (25, 8, 4)	108	−1	119	108	0
17 : (27, 8, 4)	100	102	100	100	0
18 : (50, 10, 4)	131	182	140	139	6,10
19 : (24, 10, 4)	102	−1	145	102	0
20 : (44, 10, 4)	126	−1	135	126	0
21 : (32, 12, 4)	297	−1	314	305	2,69
22 : (32, 12, 4)	297	−1	314	305	2,69
23 : (45, 14, 4)	305	−1	350	309	1,31
24 : (30, 8, 4)	199	−1	242	199	0
25 : (26, 8, 4)	141	−1	169	141	0
26 : (17, 10, 4)	65	−1	65	65	0
27 : (19, 10, 4)	107	−1	124	107	0
28 : (20, 10, 4)	121	−1	122	121	0
29 : (50, 12, 4)	202	−1	261	202	0
30 : (50, 12, 4)	202	−1	261	46	0

- Table 4:
  - The value V_Greed(1), V_Greed(50) and V_Greed(100), obtained through GREEDY(50) and GREEDY_EPC(100);
  - In case of GREEDY_EPC(100), we also mention the gap GAP_100 = (V_Greed (100) - Opt)/Opt;

Table 5 Number of states DP_EPC versus EPC_Pipeline

Instance Id: (N, M, p)	S_DP_EPC	S_DP_Prod	S_DP_BVD	CPU_DP_EPC (Sec)	$CPU_{Pipe(Sec)}$
1 : (15, 4, 4)	28	49	1	0,013	0,408
2 : (78, 10, 1)	30873	3024	15	4,026	38,736
3 : (94, 10, 1)	20370	4187	11	1,718	78,6
4 : (114, 10, 1)	9030	4760	19	1,01	73,614
5 : (99, 10, 1)	10281	4161	1	0,756	64,415
6 : (59, 10, 2)	13339	2626	1	0,895	25,965
7 : (36, 10, 2)	3082	2900	1	0,084	16,355
8 : (78, 10, 2)	10536	4404	1	0,776	57,377
9 : (57, 10, 2)	2938	1647	19	0,128	11,407
10 : (26, 8, 4)	916	1730	1	0,041	8,682
11 : (26, 10, 4)	2844	1409	11	0,063	4,792
12 : (24, 10, 4)	1933	312	1	0,045	1,077
13 : (30, 10, 4)	7916	530	1	0,173	2,533
14 : (33, 10, 4)	12201	770	1	0,246	2,867
15 : (20, 8, 4)	581	872	1	0,026	2,377
16 : (25, 8, 4)	3503	70	13	0,055	0,191
17 : (27, 8, 4)	312	2193	19	0,031	8,875
18 : (50, 10, 4)	10608	3222	16	0,6	26,089
19 : (24, 10, 4)	7995	2022	11	0,182	5,825
20 : (44, 10, 4)	492	5320	11	0,009	37,125
21 : (32, 12, 4)	10805	283	1	0,21	1,878
22 : (32, 12, 4)	10805	283	1	0,179	1,839
23 : (45, 14, 4)	22307	1725	1	1,673	13,349
24 : (30, 8, 4)	11486	573	1	0,165	2,856
25 : (26, 8, 4)	2730	1159	5	0,058	4,801
26 : (17, 10, 4)	35	629	5	0,088	1,696
27 : (19, 10, 4)	370	174	1	0,026	0,483
28 : (20, 10, 4)	545	76	19	0,027	0,169
29 : (50, 12, 4)	25020	6497	1	2,802	54,26
30 : (50, 12, 4)	25020	6497	1	2,802	61,782

- Table 5:
  - The largest number S_DP_EPC of states created by DP_EPC at a given time (i, j), together with related CPU time (in seconds) CPU_DP_EPC;
  - The largest number S_Prod of states created by DP_Ext_Prod at a given time i, together with the running time CPU_Pipe of the whole EPC_Pipeline process.
  - The largest number S_BVD of states created by DP_BVD at a given time j.

Results: We get, for a package of 30 instances generated this way, the Tables 3, 4, 5.

Comments: We notice a case when EPC_Pipeline fails in getting a feasible solution. Still, in average, its precision remains very close to optimality. As we shall see in Table 5, this small level of error will be balanced by significantly lower computational costs.

Comments: When the Max_State parameter is small, GREEDY_EPC(Max_State) often yields a failure result: it does not succeed in building a feasible solution. But, when Max_State becomes larger than 50, the failure situations happen more and more scarcely, and, when Max_State becomes larger than 50, performances of GREEDY_EPC(Max_State) become rather good, though they remain below those of EPC_Pipeline. But in any case, what has to be also taken into account is the ability of EPC_Pipeline to potentially fit with collaborative contexts, while GREEDY_EPC remains the expression of a fully centralized paradigm.

Comments: The pipe-line scheme EPC_Pipeline involves significantly less states and CPU time, for a gap almost negligible, than the exact algorithm DP-EPC. Besides, it is far more flexible, and likely to fit with some collaborative context.

7 Conclusion

We have been presenting here a bilevel dynamic programming scheme in order to solve a scheduling problem which requires synchronizing mechanisms. Many issues remain to be addressed: Extending our approach to several vehicles; Dealing with uncertainties related to H^2 production; Casting the routing issue into the decision process; Adapting our algorithms to *one line* or dynamic decision making.

References

1. Albers, S.: Energy-efficient algorithms. Commun. ACM **53**(4), 86–96 (2010)
2. Angel, E., Bampis, E., Chau, V.: Low complexity scheduling algorithms minimizing the energy for tasks with agreeable deadlines. Discret. Appl. Math. **175**, 1–10 (2014)
3. Benini, L., Bogliolo, A., De Micheli, G.: A survey of design techniques for system level dynamic power management. IEEE Trans. Very Large Scale Integr. Syst. **8**(3), 299–316 (2000)
4. Burke, A.: Batteries and ultracapacitors for electric, hybrid, and fuel cell vehicles. Proc. IEEE **95**, 806–820 (2007)
5. Chan, C.C.: The state of the art of electric, hybrid, and fuel cell vehicles. Proc. IEEE **95**, 704–718 (2007)
6. Chretienne, P., Quilliot, A.: A polynomial algorithm for the homogeneous non idling scheduling problem of unit-time independent jobs on identical parallel machines. Discret. Appl. Math 20 pages (2018)
7. Chretienne, P., Fouilhoux, P., Quilliot, A.: Anchored reactive and proactive solutions to the CPM-scheduling problem. Eur. J. Oper. Res. 261-1(67–74) (2017)
8. Grimes, C., Varghese, O., Ranjan, S.: Light, Water, Hydrogen: The Solargeneration of Hydrogen by Water Photoelectrolysis. Springer, USA (2008)

9. Irani, S., Pruhs, K.: Algorithmic problems in power management. SIGACT News **36**(2), 63–76 (2003)
10. Kara, I., Kara, B.Y., Kadri Yetis, M.: Energy minimizing vehicle routing problem. In: Andreas Dress, Yinfeng [14]. Xu, Zhu, B. (eds.), Combinatorial Optimization and Applications, pp. 62–71. Berlin, Heidelberg (2007)
11. Lajunen, A.: Energy consumption and cost-benefit analysis of hybrid and electric city buses. Transp. Res. Part C: Emerg. Technol. **38**, 1–15 (2014)
12. Licht, S.: Thermochemical and Thermal/Photo Hybrid Solar Water Splitting. Springer, New York, NY (2008)
13. Lin, C., Choy, K.L., Ho, G.T., Chung, S.H., Lam, H.: Survey of green vehicle routing problem: past and future trends. Expert Syst. Appl. **41**, 1118–1138 (2014)
14. Moon, J.-Y., Park, J.-W.: Smart production scheduling with time-dependent and machine-dependent electricity cost by considering distributed energy resources and energy storage. Int. J. Product. Res. **52** (2013)
15. Pechmann, A., Schöler, I.: Optimizing energy costs by intelligent production scheduling. In: Hesselbach, J., Herrmann, C. (ed.), Globalized Sustainability in Manufacturing, Berlin, Heidelberg, pp. 293–298 (2011)
16. Waraich, R., Galus, M., Dobler, C., Balmer, M., Andersson, G., Axhausen, K.: Plug-in hybrid electric vehicles and smart grid: investigations based on a micro simulation. Transp. Res. Part C Emerg. Technol. **28** (2014)

Reducing the First-Type Error Rate of the Log-Rank Test: Asymptotic Time Complexity Analysis of An Optimized Test's Alternative

Lubomír Štěpánek, Filip Habarta, Ivana Malá, and Luboš Marek

Abstract Comparing two time-event survival curves representing two groups of individuals' evolution in time is relatively usual in applied biostatistics. Although the log-rank test is the suggested tool for facing the above-mentioned problem, there is a rich statistical toolbox used to overcome some of the log-rank test properties. However, all of these methods are limited by relatively rigorous statistical assumptions.In this study, we introduce a new robust method for comparing two time-event survival curves. We briefly discuss selected issues of the log-rank test's robustness and analyze a bit more some of the properties and mostly asymptotic time complexity of the proposed method. The new method models individual time-event survival curves in a discrete combinatorial way as orthogonal monotonic paths, enabling direct estimation of the p-value as it was originally defined. Finally, using simulated time-event data, we check the robustness of the introduced method compared to the log-rank test.Based on the theoretical analysis and simulations, the introduced method seems to be a promising alternative to the log-rank test, particularly when comparing two time-event curves regardless of any statistical assumptions with a generally low risk of the first type error.

L. Štěpánek (✉)
Department of Statistics and Probability, Faculty of Informatics and Statistics, University of Economics, nám. W. Churchilla 4, 130 67 Prague, Czech Republic & Institute of Biophysics and Informatics, First Faculty of Medicine, Charles University, Salmovská 1, Prague, Czech Republic
e-mail: lubomir.stepanek@vse.cz; lubomir.stepanek@lf1.cuni.cz

F. Habarta · I. Malá · L. Marek
Department of Statistics and Probability, Faculty of Informatics and Statistics, University of Economics, nám, W. Churchilla 4, 130 67 Prague, Czech Republic
e-mail: filip.habarta@vse.cz

I. Malá
e-mail: malai@vse.cz

L. Marek
e-mail: marek@vse.cz

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_15

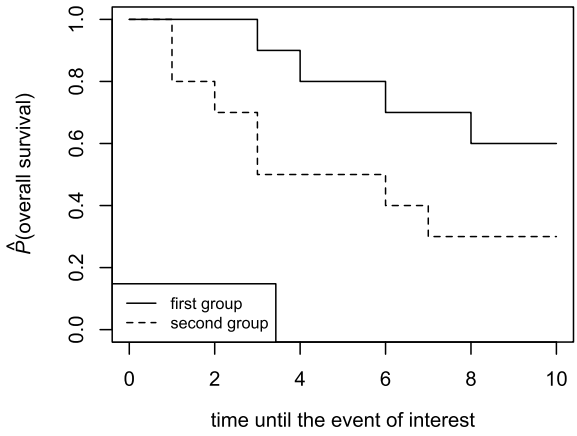


Fig. 1 Two time-event survival curves in a survival plot

1 Introduction

In survival analysis, the response variable is usually two-dimensional, since it takes into account both the time of the event of our interest and whether the event (or the censoring) even occurred. More than intuitively, such a target variable suggests being plotted in a two-dimensional plot. As usual, while a number of subjects who do not evinced the event of interest to all subjects is plotted on a vertical axis at a given time point, the time points where the event occurred are aligned with the horizontal axis, see also Fig. 1 That is the way how Kaplan-Meier estimators are commonly illustrated [1]. Therefore, the survival curve as a response variable could be represented as a monotonic orthogonal path, i.e., a polygonal path of a finite number of horizontal and vertical segments, in the Cartesian two-dimensional chart. Since such a variable deals both with the events of interest and their times, it is ordinarily called the time-event (survival) curve.

Whenever two or more time-event survival curves, describing evolution of events in time within two groups of individuals, are to be compared, several well-established methods could be used. A classical log-rank test could solve the problem, when there are only two groups supposed to be compared [2]. Assuming some special settings, particularly when the time-event survival curves are constructed using data that are not censored, i.e. the data fully describe all events of interest occurred in the groups, then a simple Wilcoxon rank-sum test might be applied. If more than two groups are to be compared, the problem could be battled using either a score-rank test, or even a Cox proportional hazards model [3]. All the approaches mentioned above may be performed in various software, including R language and environment [4], such that a pure R package `stats` or a package `survival` [5] could be employed to do the job. Nevertheless, each one of the described methods has its limitations, and its application is determined by meeting relatively rigorous statistical assumptions.

Application of the log-rank test that compares two non-crossing time-event survival curves (similar as plotted in Fig. 1) is limited mostly by assuming the fact that

censoring (i) should not induce anyhow the observed events and (ii) is equally likely to occur in both the groups. What is more, the counts of events of interest should be large enough to satisfy asymptotic properties of χ^2 distribution and fulfill the central limit theorem to let the log-rank test statistic follow an asymptotically normal distribution. That means an incidence of the events of interest in each group across all the time points should be neither too small nor too large.

To overcome the limitations of the classical log-rank test, several diverse modifications of the log-rank test were published to either increase efficiency of the test, or its robustness against violation of the statistical assumptions, or both. Whereas Kong (1997) in [6] adjusted the log-rank test efficiency by improving the hazard functions, i.e. functions of rates of events based on fixed proportions of the events in the past, Song et al. (2008) dived deeper into covariate matrix decomposition, by which they derived formulas estimating minimal sample sizes that enables a valid usage of the log-rank test [7]. Several authors such as Peto and Peto (1972) [8], Yang and Prentice (2010) [9], and Li (2018) [10], suggested the usage of weights of individual observations, usually lower weights for later events when there the numbers of observations tend to be not so high; by this they improve the validity of the log-rank test outputs.

There are also articles handling with exact discrete calculations when compare two survival curves which is more similar to our proposed approach. Thomas (1975) simplified the computations by fixing total numbers in the compared groups [11]. The algorithm was improved a bit computationally by Mehta et al. (1985) [12]. Finally, Heinze et al. (2003), similarly asymptotic approaches above, incorporated a weighting scheme into the calculations to increase significance of earlier observations [13].

Studies that go deeper into asymptotic complexities of the statistical inference test, particularly the exact ones that exhaustively compute over a polynomial universe, are generally missing. Some significant pieces of related knowledge focused on complexity of classic but robust and computationally-hard inference tests are discussed by Mosler (2002) [14], Smolinski et al. in (2008) [15], and Kulikov et al. (2014) [16].

Vast majority of the papers listed above work with a hazard function, which is event of interest rate in a given time point conditional on overall survival rate until the time point or they assume constant total numbers of subjects in all the compared groups. Unlike them, in this proceeding, besides a brief discussion on limitations of the log-rank test, we model the time-event survival curves using a discrete combinatorial approach, considering the survival curves to be orthogonal monotonic paths on a plane of two-dimensional plot (as shown in Figs. 1 and 2), and taking into account their mutual "Manhattan" grid distances. That indicates how easily the p-value of this modified log-rank test could be calculated using its original statistical definition as a conditional probability of observing data of given properties. Then we analyse asymptotic time complexity of algorithmic approaches behind the proposed method. We also briefly discuss the possible relationship between the two-dimensional surface bounded by two non-crossing survival curves in the plot and the test's p-value. Finally, using simulations of artificial survival curves, the first type errors as rates of detection the cases, when similar curves are supposed to be different, are esti-

Table 1 Numbers of the events of interest in both groups at time t_j

	Event of interest at the event time t_j		
Group	Yes	No	Total
1	$d_{1,j}$	$r_{1,j} - d_{1,j}$	$r_{1,j}$
2	$d_{2,j}$	$r_{2,j} - d_{2,j}$	$r_{2,j}$
Total	d_j	$r_j - d_j$	r_j

mated for both the log-rank test and our proposed alternative, mutually compared and discussed within the frame of the robustness of the methods.

2 Principles, Assumptions and Limitations of the Log-Rank Test

Firstly, we gently introduce principles of the log-rank test, by which we can better understand its assumptions and limitations.

2.1 Principles of the Log-Rank Test

Let's assume two groups of individuals (marked by indices 1, and 2, respectively) and $k \in \mathbb{N}$ distinct event times. At each event time, we can construct a 2×2 contingency table and compare the event rates between the two groups. Let the $(t_1, t_2, \ldots, t_k)^T$ be an ordered tuple of the event time points, then for the j-th event time t_j, such that $j \in \{1, 2, 3, \ldots, k\}$, we can construct the (contingency) table Table 1. At j-th event time, there are $d_{1,j}$ and $d_{2,j}$ individuals who experienced the events in the group 1 and 2, respectively, and $r_{1,j}$ and $r_{2,j}$ subjects at risk (who have not yet had the event or been censored) in the groups 1 and 2, respectively, see Table 1.

The log-rank test checks the null hypothesis H_0 that both groups have identical hazard functions, i.e. that rates of the events of interest in time conditional on fixed rates in the past are the same. Under the null hypothesis H_0, the observed numbers of the events could be considered as random variables $D_{1,j}$ and $D_{2,j}$ following a hypergeometric distribution with parameters $(r_j, r_{i,j}, d_j)$ for both $i \in \{1, 2\}$. Thus, the expected value of the variable $D_{i,j}$ is $\mathbb{E}(D_{i,j}) = r_{i,j}\frac{d_j}{r_j}$ and variance is $\mathrm{var}(D_{i,j}) = \frac{r_{1,j}r_{2,j}d_j}{r_j^2}\left(\frac{r_j-d_j}{r_j-1}\right)$ for both $i \in \{1, 2\}$. For all $j \in \{1, 2, 3, \ldots, k\}$ we can compare the observed numbers of events of interest, $d_{i,j}$, to their expected values $\mathbb{E}(D_{i,j}) = r_{i,j}\frac{d_j}{r_j}$, under H_0. So, the test statistic for both $i \in \{1, 2\}$ is finally

$$\chi^2_{\text{log-rank}} = \frac{\left(\sum_{j=1}^{k} d_{i,j} - \mathbb{E}(D_{i,j})\right)^2}{\sum_{j=1}^{k} \text{var}(D_{i,j})} = \frac{\left(\sum_{j=1}^{k} d_{i,j} - r_{i,j}\frac{d_j}{r_j}\right)^2}{\sum_{j=1}^{k} \frac{r_{1,j}r_{2,j}d_j}{r_j^2}\left(\frac{r_j-d_j}{r_j-1}\right)}, \tag{1}$$

which follows under H_0 a χ^2 distribution with 1 degree of freedom, $\chi^2_{\text{log-rank}} \sim \chi^2(1)$. For feasible large r_j, at least $r_j \geq 30$, a square root of $\chi^2_{\text{log-rank}}$ follows a standard normal distribution, $\sqrt{\chi^2_{\text{log-rank}}} \sim \mathcal{N}(0, 1^2)$.

2.2 Some of the Assumptions and Limitations of the Log-Rank Test

Firstly, censoring is assumed not to affect anyhow the occurrence of event of interest, and the proportion of censored data are supposed being of nearly equal size in both the groups, as well. Otherwise, the test statistic $\chi^2_{\text{log-rank}}$ calculated using (1) either for $i = 1$, or for $i = 2$, respectively, could be biased and therefore mutually different. That may affect the interpretability, i.e. the robustness of the log-rank test applied on such data [17].

Then, since the test statistic $\chi^2_{\text{log-rank}}$ follows a χ^2 distribution, the initial total number of individuals r_0 and the number of all event times k should be large enough. Analogously but inversely, whenever the numbers of individuals d_j experiencing the event of interest are generally large (relatively to r_j), than both the numerator and denominator of the fraction in the formula (1) is relatively small, too, and, consequently, one could expect that the $\chi^2_{\text{log-rank}}$ statistic (or the derived $\sqrt{\chi^2_{\text{log-rank}}}$ statistic) does not fulfil its assumed asymptotic properties, and its estimate could be thus biased. That might influence both the robustness and the power of the log-rank test when applied to data of such limitations [18].

By researching the denominator of the Eq. (1) a bit deeper, we can realize the test statistic $\chi^2_{\text{log-rank}}$ is the highest when the denominator $\sum_{j=1}^{k} \text{var}(D_{i,j})$ is as low as possible given the values $d_{i,j}$ and $r_{i,j}$ for all $i \in \{1, 2\}$ and $j \in \{1, 2, 3, \ldots, k\}$. It is worth mentioning this holds just when the proportions $\frac{r_{1,j}}{r_j} = \frac{r_{1,j}}{r_{1,j}+r_{2,j}}$ and $\frac{r_{2,j}}{r_j} = \frac{r_{2,j}}{r_{1,j}+r_{2,j}}$ are both constant (and mutually different enough) across all the time points $(t_1, t_2, \ldots, t_k)^T$, and then the log-rank test is the most powerful; i.e. in other words, its ability to reject the null hypothesis H_0, claiming the survival curves are equivalent, when they are in fact different, is maximal possible. That used to be the most usual issue that may decrease the power of the log-rank test. The mentioned proportions are typically not constant when the time event curves change a lot their mutual distance across the time points or when they even cross themselves one or more times.

Consequently, the power of the log-rank test may be decreased by any deviations from the constant values of the proportions $\frac{r_{1,j}}{r_j}$, and $\frac{r_{2,j}}{r_j}$, respectively.

3 Introduction of an Assumption-Free Alternative to the Log-Rank Test

Within this section, we introduce an assumption-free alternative to the log-rank test. The alternative algorithm for two time-event curves comparison is based on a discrete combinatorial calculation of possible states (i.e. all possible time-event curves) that would be theoretically obtained and that are at least as extreme as the original two survival curves. This approach corresponds to an original definition of a p-value as a probability of obtaining data at least as extreme as the data currently observed, assuming that the null hypothesis is true (i.e. the observed survival curves are not statistically different).

All the possible states could be considered as monotonic orthogonal paths in the two-dimensional chart of two original survival curves, excluding (for simplicity) the crossing curves. By calculating (or estimating) the numbers of all the paths at least at extreme as the plotted two curves, i.e. all the paths such that one is above the first observed one and the other is below the second observed one, we get a point estimate of the p-value as a proportion of all pairs of orthogonal paths contradicting the same way or even more to the observed survival curves. Or in other words, as a proportion of all pairs of orthogonal paths that are at least as distant one from the other than the original two time-event curves.

3.1 Principle of the Proposed Assumption-Free Alternative to the Log-Rank Test

Again, let two groups of individuals (marked by indices 1, and 2, respectively) to be compared and $k \in \mathbb{N}$ distinct event times when events of interest could occur. Let the $(t_1, t_2, \ldots, t_k)^T$ be an ordered tuple of the event times. At each event time, we can compute the number of individuals who experienced the event at the j-th event time t_j for both groups, similarly to the construction of contingency tables, as shown in table Table 1. By repeating this approach k times, consequently, once we get the proportions of subjects at risk, $\frac{r_{1,j}}{r_j}$, and $\frac{r_{2,j}}{r_j}$, respectively, for each event time t_j, we could plot the time-event survival curves based on the proportions of individual in risk $\frac{r_{1,j}}{r_j}$, and $\frac{r_{2,j}}{r_j}$ similarly to Fig. 1.

For simplicity, the survival curves are assumed not to cross themselves. More technically spoken, it for each j-th event time t_j holds

$$\frac{r_{1,j}}{r_j} \geq \frac{r_{2,j}}{r_j}, \tag{2}$$

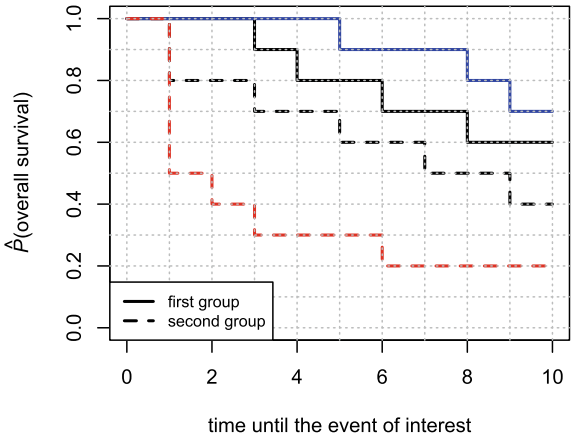


Fig. 2 Two original time-event survival curves in a survival plot (black lines) and an example of a pair of monotonic orthogonal paths such that one is above (blue solid line, $i = 1$) the upper original survival curve and the second one is below (red dashed line, $i = 2$) the lower original survival curve

as illustrated in Fig. 1. By adding a grid into the Fig. 1, we get Fig. 2, which is a bit closer to an idea of calculating (or estimating) a number of monotonic orthogonal paths starting at the proportion of subjects at risk $\frac{r_{i,0}}{r_0} = 1$ and ending – after k event times – at the proportion of subjects at risk $\geq \frac{r_{i,k}}{r_k}$ (one of such possible paths is the blue line for $i = 1$ in Fig. 2) $\leq \frac{r_{i,k}}{r_k}$ (similarly to the red line for $i = 2$ in Fig. 2).

Let $N_{(1,k,u,v)}$ stands for the number of all orthogonal paths (respecting the grid, i.e. all segments of such a path are parallel to horizontal or vertical lines of the grid and its edges are aligned to grid points) starting at the proportion 1 (left upper corner of the Fig. 2) and ending after k event times at the proportion of subjects at risk $\frac{u}{v}$ (a point with coordinates $[k, \frac{u}{v}]$ in Fig. 2). Eventually, let $N^{+}_{(1,k,u,v)}$ be a number of all orthogonal paths starting at the proportion 1, going above the 1-st survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk $\geq \frac{u}{v}$. Analogously, let $N^{-}_{(1,k,u,v)}$ be a number of all orthogonal paths starting at the proportion 1, going below the 2–nd survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk $\leq \frac{u}{v}$ after k event times. The numbers $N^{+}_{(1,k,u,v)}$ and $N^{-}_{(1,k,u,v)}$ could be computed perhaps exhaustively in a combinatorial way (this is an open problem) or could definitely be estimated by numerical simulations.

Let us define a null hypothesis H_0 that claims the original (observed) survival curves are not significantly different. On of the tricky part on the proposed method is that, since we do not need any more initial assumptions for this testing, we also do not require modelling a null distribution. The p-value, as mentioned above, is the probability of obtaining data (expected survival curves described as monotonic orthogonal paths in the survival plot) at least as extreme as the data currently observed (the two original survival curves), assuming that the null hypothesis H_0 is correct. Following the definition of the p-value and marking it as p, we get

$$p = p - \text{value}$$
$$p = P(\text{getting data at least as extreme as the observed}|H_0)$$
$$p = P\left(\frac{N^{+}_{1,k,r_{1,k},r_k} \cdot N^{-}_{2,k,r_{2,k},r_k}}{\left(\sum_{j=0}^{r_k} N_{k,j,r_k}\right)^2 - N_{\text{cc}},}\right) \tag{3}$$

where N_{cc} is a number of pairs of survival curves crossing each other. Again, the number N_{cc} can be calculated probably either using a discrete combinatorial analysis, or be numerically simulated (which is far easier).

In comparison with the term in the denominator of the Eq. (3), the number of pairs of survival curves depicted by the numerator can not include any crossing curves. Since we assume all curves ending in the proportion $\frac{r_{1,k}}{r_k}$ or greater, and all curves ending in the proportion $\frac{r_{2,k}}{r_k}$ or lower, considering that $\frac{r_{1,j}}{r_j} \geq \frac{r_{2,j}}{r_j}$ for each $j \in \{1, 2, 3, \ldots, k\}$ as stated in (2), thus, since $\frac{r_{1,k}}{r_k} \geq \frac{r_{2,j}}{r_j}$ for all time points, there are no pairs of crossing curves taken into account in the numerator of (3). The curves could tangentially meet themselves (in case of $=$) or run one above the other (in case of $>$), but could not cross each other.

3.2 A Brief Analysis of Surface Bounded by Two Non-crossing Survival Curves and the Test's p-value

Surfaces above the first, upper survival curve (let us mark it as S_1^+) and below the second, bottom curve (let us mark it as S_2^-) in Fig. 2 suggest investigating on how are the surfaces related to the p-value of the test.

By following the first impression, when S stands for a surface of the whole canvas of the chart in Fig. 2, it seems that p-value is proportional to the term $\frac{S_1^+ + S_2^-}{S}$. However, the relationship between the p-value and the surfaces is more complex and not so straightforward. The numbers of all orthogonal paths in some dedicated surface, let us assume e.g. $N^+_{(1,k,u,v)}$, is not proportional to the size of the surface. As a sketch of a proof by contradiction, let us suppose we are to calculate the number of $N^-_{(1,k,0,v)}$ curves below a horizontal curve crossing the point $[k, v]$. Then, simply using combinatorial rules, we realize that $N^-_{(1,k,0,v)} = \binom{k+v}{k}$. However, if we now want to calculate the number of $N^-_{(1,k,0,2v)}$ curves below a horizontal curve crossing the point $[k, 2v]$, we get that $N^-_{(1,k,0,2v)} = \binom{k+2v}{k}$. Whereas the proportion of the surfaces below the two lines crossing the points $[k, 2v]$ and $[k, v]$ is equal to 2, the proportion of the numbers of the paths is in general much greater than 2, since generally $\frac{\binom{k+2v}{k}}{\binom{k+v}{k}} \gg 2$. Thus, $\frac{N^+_{1,k,r_{1,k},r_k}}{N^-_{2,k,r_{2,k},r_k}} \neq \frac{S_1^+}{S_2^-}$ in general and the p-value is not (!) proportional to the term $\frac{S_1^+ + S_2^-}{S}$.

3.3 Approaches on Calculation the p-value of the Proposed Alternative to the Log-Rank Test

The terms such as $N^{+}_{1,k,r_{1,k},r_k}$, $N^{-}_{2,k,r_{2,k},r_k}$, $\sum_{j=0}^{r_k} N_{k,j,r_k}$, and N_{cc}, respectively, in Eq. (3) could by estimated either numerically by re-sampling, or calculated exhaustively. A fully analytical approach is under current research.

3.3.1 Re-Sampling Approach

All the terms such as $N^{+}_{1,k,r_{1,k},r_k}$, $N^{-}_{2,k,r_{2,k},r_k}$, $\sum_{j=0}^{r_k} N_{k,j,r_k}$, and N_{cc}, respectively, in equation (3) could be numerically estimated by *re-sampling* approach. Let us assume we got two non-crossing survival curves similarly to the plot in Fig. 1, so that we know the values $k, r_{1,j}, r_{2,j}, d_{1,j}$, and $d_{2,j}$ for all $j \in \{1, 2, 3, \ldots, k\}$. Let us suppose we generate n pairs of survival curves. Then, let $N(\forall+, \forall-)(n)$ be the number of all pairs such that one of the curves is completely above the first original curve and the other is completely below the second original curve (and, thus, they do not cross each other), in all n generated pairs. Let $N(\text{non-crossing})(n)$ be the number of all pairs such that the curves of the pair do not cross each other, in all n generated pairs. Then we can simply derive that

$$N^{+}_{1,k,r_{1,k},r_k} \cdot N^{-}_{2,k,r_{2,k},r_k} = \lim_{n\to\infty} N(\forall+, \forall-)(n)$$

$$\left(\sum_{j=0}^{r_k} N_{k,j,r_k}\right)^2 - N_{\mathrm{cc}} = N(\text{non-crossing})(n),$$

and, consequently, by replacing in Eq. (3)

$$\hat{p} = \lim_{n\to\infty} \frac{N(\forall+, \forall-)(n)}{N(\text{non-crossing})(n)}.$$

By this *re-sampling* approach, we can obtain for reasonably large $n \in \mathbb{N}$ an unbiased estimate of p-value in Eq. (3) of the proposed alternative to the log-rank test. The algorithm is also described in Algorithm 1.

Algorithm 1: Re-sampling approach on how to obtain for reasonably large $n \in \mathbb{N}$ an unbiased estimate of p-value in Eq. (3) of the proposed alternative to the log-rank test.

Data: two non-crossing survival curves
Result: an unbiased estimate of p-value in Eq. (3) of the proposed alternative to the log-rank test

```
k, r_{1,j}, r_{2,j}, d_{1,j}, d_{2,j}      // parameters of the original two survival curves ;
n                                          // number of repetitions;
N(∀+, ∀−)(0) = 0                           // number of all pairs such that one of the curves is completely above the first original curve and the other is completely below the second original curve;
N(non-crossing)(0) = 0                     // number of all pairs such that the curves of the pair do not cross each other;
for j = 1 : n do
    generate a pair of two survival curves;
    if the curves of the pair do not cross each other then
        N(non-crossing)(j) = N(non-crossing)(j) + 1;
        if one of the curves is completely above the first original curve and the other is completely below the second original curve then
            N(∀+, ∀−)(j) = N(∀+, ∀−)(j) + 1;
end
calculate an estimate of p-value as p̂ = N(∀+,∀−)(n) / N(non-crossing)(n) ;
```

3.3.2 Exhaustive Approach

Let us again assume we got two non-crossing survival curves similarly to the plot in Fig. 1, so that we know the values $k, r_{1,j}, r_{2,j}, d_{1,j}$, and $d_{2,j}$ for all $j \in \{1, 2, 3, \ldots, k\}$. The exhaustive, greedy approach is based on grid search for all possible pairs of survival curves such that one of the curves is completely above the first original curve and the other is completely below the second original curve (and, thus, they do not cross each other). In case the exhaustive approach is finished successfully, one could obtain more confident estimate of p-value in Eq. (3) of the proposed alternative to the log-rank test than in case of the numerical re-sampling.

Given that a value k for all considered time event points, $r_{1,k}$ for a number of subjects at risk in group 1 after k time points, $r_{2,k}$ for a number of subjects at risk in the group 2 after k time points, and r_k for a total number of subjects at risk in both the groups (1 and 2) after k time points are known before any calculations of the p-value; thus, all four terms, i.e.

(i) the term $N_{(1,k,u,v)}$ standing for the number of all orthogonal paths starting at the proportion 1 and ending after k event times at the proportion of subjects at risk $\frac{u}{v}$,

(ii) the term $N^{+}_{(1,k,u,v)}$ as a number of all orthogonal paths starting at the proportion 1, going above the 1–st survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk $\geq \frac{u}{v}$ after k event times,
(iii) the term $N^{-}_{(1,k,u,v)}$ as a number of all orthogonal paths starting at the proportion 1, going below the 2–nd survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk $\leq \frac{u}{v}$ after k event times,
(iv) the term N_{cc} as a number of pairs of survival curves crossing each other,

could be exactly calculated or numerically simulated. Furthermore, using the formula (3) the p-value of the proposed alternative to the log-rank test is uniquely determined. So, the exhaustive approach returns an exact value of the p-value.

Before we start with derivations of asymptotic time complexity of the exhaustive approach, in order to clarify all the logic behind the approach, we sketch the principle of the exhaustive approach for computations of $N^{+}_{(1,k,u,v)}$ and $N^{-}_{(1,k,u,v)}$ in Algorithms 2 and 3.

Algorithm 2: Exhaustive approach on how to obtain an exact estimate of the $N^{+}_{(1,k,u,v)}$ term for p-value in Eq. (3) of the proposed alternative to the log-rank test.

Data: two non-crossing survival curves
Result: an exact calculation of the $N^{+}_{(1,k,u,v)}$ term for p-value in Eq. (3) of the proposed alternative to the log-rank test

```
k, r_{1,j}, r_{2,j}, d_{1,j}, d_{2,j}      // parameters of the original two survival curves ;
N^+_{(1,k,u,v)} = 0                        // number of all orthogonal paths starting at the proportion 1, going above the 1-st survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk ≥ u/v after k event times;
i = v;
while i ≥ u do
    N_{(1,k,j,v)} = number of all orthogonal paths starting at the proportion 1 and ending after k event times at the proportion of subjects at risk i/v
    N^+_{(1,k,u,v)} = N^+_{(1,k,u,v)} + N_{(1,k,u,v)}
    i = i − 1
end
N^+_{(1,k,u,v)};
```

Despite the linear calculations introduced by `while` loops in Algorithms 2 and 3, an asymptotic time complexity of the terms $N^{+}_{(1,k,u,v)}$ and $N^{-}_{(1,k,u,v)}$ computations is more complex, since computations of the term $N_{(1,k,u,v)}$ are exponential as indicated in the Sect. 3.2 about the brief analysis of surface bounded by two non-crossing survival curves.

Algorithm 3: Exhaustive approach on how to obtain an exact estimate of the $N^{-}_{(1,k,u,v)}$ term for p-value in Eq. (3) of the proposed alternative to the log-rank test.

Data: two non-crossing survival curves
Result: an exact calculation of the $N^{-}_{(1,k,u,v)}$ term for p-value in Eq. (3) of the proposed alternative to the log-rank test

```
k, r_{1,j}, r_{2,j}, d_{1,j}, d_{2,j}      // parameters of the original two survival curves ;
N^-_{(1,k,u,v)} = 0                        // number of all orthogonal paths starting at the proportion 1, going below the 2-nd survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk ≤ u/v after k event times;
i = 0;
while i ≤ u do
    N_{(1,k,j,v)} = number of all orthogonal paths starting at the proportion 1 and ending after k event times at the proportion of subjects at risk i/v
    N^-_{(1,k,u,v)} = N^-_{(1,k,u,v)} + N_{(1,k,u,v)}
    i = i + 1
end
N^-_{(1,k,u,v)};
```

Since the exhaustive approach is greedy, so that one could expect a large asymptotic time complexity, we enumerated worst-case scenarios estimates of all terms such as $N^{+}_{1,k,r_{1,k},r_k}$, $N^{-}_{2,k,r_{2,k},r_k}$, $\sum_{j=0}^{r_k} N_{k,j,r_k}$, and N_{cc}, respectively, in Eq. (3).

Since $N^{+}_{(1,k,r_{1,k},r_k)}$ (or $N^{-}_{(2,k,r_{2,k},r_k)}$) is a number of all orthogonal paths starting at the proportion 1, going above (or below) the 1–st (or the 2-nd) survival curve or tangentially meeting it (without crossing it) and ending at the proportion of subjects at risk $\geq \frac{u}{v}$, number of such paths could not be larger than a number of all monotonic orthogonal paths in a rectangle of size $k \times d_{1,k}$ (or $k \times d_{2,k}$). Then,

$$N^{+}_{1,k,r_{1,k},r_k} \leq \sum_{j=0}^{d_{1,k}} \binom{k+j}{k} = \binom{d_{1,k}+k+1}{k+1}$$

$$N^{-}_{2,k,r_{2,k},r_k} \leq \sum_{j=0}^{r_{2,k}} \binom{k+j}{k} - \sum_{j=0}^{d_{2,k}} \binom{k+j}{k} =$$

$$= \binom{r_{2,k}+k+1}{k+1} - \binom{d_{2,k}+k+1}{k+1}.$$

Number of all monotonic orthogonal paths in the grid, $\sum_{j=0}^{r_k} N_{k,j,r_k}$, is by assuming (for simplicity) $r_{1,k} = r_{2,k}$ similarly

$$\sum_{j=0}^{r_k} N_{k,j,r_k} \leq \sum_{j=0}^{r_{2,k}} \binom{k+j}{k} = \binom{r_{2,k}+k+1}{k+1}.$$

Since parts of crossing curves in a pair could be rearranged such that the crossing segments could be "re-coloured" eventually, i.e. switched so that the curves only tangentially meet each other keeping them monotonic, we can assume that $N_{\mathrm{cc}} \ll \sum_{j=0}^{r_k} N_{k,j,r_k}$.

Putting all the derivations together, we can estimate an upper estimate $\Theta(\bullet)$ of all the monotonic orthogonal paths' grid searching by the formula

$$\begin{aligned}
\Theta(\bullet) = \quad & \Theta\left(N^{+}_{1,k,r_{1,k},r_k}\right) + \Theta\left(N^{-}_{2,k,r_{2,k},r_k}\right) + \\
& + \Theta\left(\sum_{j=0}^{r_k} N_{k,j,r_k}\right) + \Theta\left(N_{\mathrm{cc}}\right) \qquad = \\
= \quad & \Theta\left(\binom{d_{1,k}+k+1}{k+1}\right) + \\
& + \Theta\left(\binom{r_{2,k}+k+1}{k+1} - \binom{d_{2,k}+k+1}{k+1}\right) + \\
& + \Theta\left(\binom{r_{2,k}+k+1}{k+1}\right) + \\
& + \Theta(0) \qquad = \\
= \quad & \Theta\left((k+d_{1,k}/2)^{d_{1,k}}\right) + \\
& + \Theta\left((k+r_{2,k}/2)^{r_{2,k}} - (k+d_{2,k}/2)^{d_{2,k}}\right) + \\
& + \Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right).
\end{aligned}$$

Since we assume $r_k > r_{1,k} \approx r_{2,k}$ and there is $r_{1,k} \geq d_{1,k}$ and $r_{2,k} \geq d_{2,k}$, then

$$\Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right) < \Theta\left((k+r_k/2)^{r_k}\right)$$

and

$$\Theta\left((k+d_{1,k}/2)^{d_{1,k}}\right) < \Theta\left((k+r_{1,k}/2)^{r_{1,k}}\right) < \Theta\left((k+r_k/2)^{r_k}\right)$$

and also

$$\begin{aligned}\Theta\left((k+r_{2,k}/2)^{r_{2,k}}-(k+d_{2,k}/2)^{d_{2,k}}\right) &= \Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right)+\Theta\left((k+d_{2,k}/2)^{d_{2,k}}\right)\\ &< \Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right)+\Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right)\\ &< \Theta\left((k+r_k/2)^{r_{2,k}}\right)+\Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right)\\ &< 2\Theta\left((k+r_k/2)^{r_{2,k}}\right).\end{aligned}$$

Assuming that a linear multiplier larger than one does not significantly increases the worst-case asymptotic time complexity, i.e. that for instance, $\Theta\left((k+r_k/2)^{r_{2,k}}\right) \approx 2\Theta\left((k+r_k/2)^{r_{2,k}}\right) \approx 3\Theta\left((k+r_k/2)^{r_{2,k}}\right) \approx 4\Theta\left((k+r_k/2)^{r_{2,k}}\right)$ etc., finally, we get

$$\begin{aligned}\Theta(\bullet) = &\quad \Theta\left((k+d_{1,k}/2)^{d_{1,k}}\right)+\\ &+\Theta\left((k+r_{2,k}/2)^{r_{2,k}}-(k+d_{2,k}/2)^{d_{2,k}}\right)+\\ &+\Theta\left((k+r_{2,k}/2)^{r_{2,k}}\right) \qquad <\\ < &\quad \Theta\left((k+r_k/2)^{r_k}\right)+\\ &+2\Theta\left((k+r_k/2)^{r_k}\right)+\\ &+\Theta\left((k+r_k/2)^{r_k}\right) \qquad =\\ = &\quad 4\Theta\left((k+r_k/2)^{r_k}\right) \qquad \approx\\ \approx &\quad \Theta\left((k+r_k/2)^{r_k}\right).\end{aligned}$$

Thus, the final worst-case scenario's asymptotic time complexity of p-value exhaustive calculation using the proposed method alternative to the log-rank test is equal to $\Theta(\bullet) = \Theta\left((k+r_k/2)^{r_k}\right)$.

In comparison to the novel method's time complexity, the log-rank test's $\chi^2_{\text{log-rank}}$ statistic based on Eq. (1) is significantly simpler, considering its asymptotic time complexity. Both by inspecting the numerator and denominator of the fraction in (1), we can see the calculation is based only on two summations of k terms, so the asymptotic time complexity of the log-rank test's $\chi^2_{\text{log-rank}}$ statistic as about $\Theta(k)$, where k is a number of time points where the event of interest may occur. However, the robustness of the classic log-rank test is limited compared to the proposed method, as is showed in Sect. 4 dedicated to a simulation study depicting some of the robust properties of the method.

3.3.3 Analytical Approach

At the current moment, a fully analytical approach on how to calculate the term $N^+_{1,k,r_{1,k},r_k}$, $N^-_{2,k,r_{2,k},r_k}$, $\sum_{j=0}^{r_k} N_{k,j,r_k}$, and N_{cc}, respectively, in Eq. (3) is an open problem and requires authors' ongoing research.

4 Simulation Study

We compared the log-rank test and the assumption-free method proposed above by simulating many pairs of random non-crossing curves, assuming the curves in pairs are not significantly different. The comparisons were made for several scenarios differing in the number of time points taken into account. Then, we calculated the first type error rates for each of the scenarios, i.e. rates of the situations, when the inference test (either the log-rank test or the new method) claims that two statistically similar survival curves are (falsely) detected as different. Finally, assuming that the more robust the method is, the lower value of the first type error it should return, we could detect whether the novel method is more robust than the traditional log-rank test.

The simulation study was performed using R programming language and environment [4]. There is more on numerical applications of R programming language to various fields in [19–24].

When generating the pairs of survival curves, we applied the following negatively exponential survival function,

$$s(t) = \sigma\left(e^{-\frac{5+\varepsilon}{200}t}\right)$$

where ε is a random noise term that follows a standard normal distribution, i.e. $\varepsilon \sim \mathcal{N}(0, 1^2)$, and $\sigma(\bullet)$ is a function rounding its argument to the nearest integer, e.g. $\sigma(4.3) = 4$, $\sigma(4.5) = 5$ or $\sigma(5.8) = 6$ (see Fig. 3).

There were $n = 1000$ pairs of significantly non-different survival curves generated in total, and within each pair, the curves were compared using both the log-rank test and the above-proposed method. This was repeated for each scenario, i.e. for each of the different number of considered time points. Numbers of considered time points follow the values of $k \in \{10, 20, 50, 100, 150, 200\}$ which feasibly respects the denominator in the exponent of the survival function $s(t) = \sigma\left(e^{-\frac{5+\varepsilon}{200}t}\right)$ as intro-

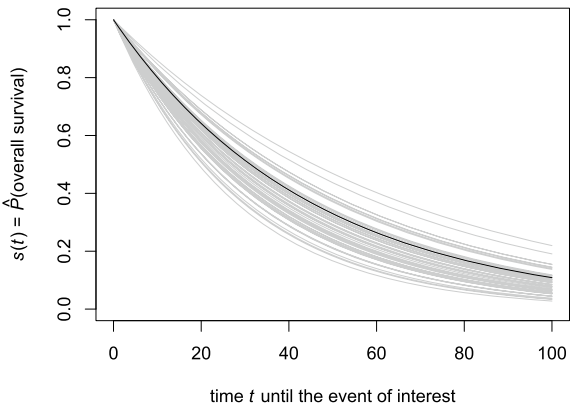


Fig. 3 An example of a group of negatively exponential survival functions following the formula $\sigma^{-1}(s(t)) = \sigma^{-1}\left(e^{-\frac{5+\varepsilon}{200}t}\right) = e^{-\frac{5+\varepsilon}{200}t}$ for different random values of $\varepsilon \sim \mathcal{N}(0, 1^2)$

Table 2 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 10$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases p-value ≤ 0.05	51	42
The first type error rate estimate	0.051	0.042

Table 3 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 20$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases $p - \text{value} \leq 0.05$	49	38
The first type error rate estimate	0.049	0.038

Table 4 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 50$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases $p - \text{value} \leq 0.05$	52	36
The first type error rate estimate	0.052	0.036

duced and described above. The fact the curves in each pair are significantly non-different was checked by the traditional log-rank test; whenever a pair consists of two statistically different survival curves, it was replaced by a newly generated one with statistically different curves.

By summing up numbers of cases where p-value was lower than or equal to 0.05 regardless of the method, we got the point estimates of the first type error frequencies as illustrated in Table 2 for $k = 10$ time points, in Table 3 for $k = 20$ time points, in Table 4 for $k = 50$ time points, in Table 5 for $k = 100$ time points, in Table 5 for $k = 150$ time points, in Table 7 for $k = 200$ time points, respectively.

Point estimates of the first type error rates for the log-rank test and the proposed method are also compared for all scenarios differing in considered time points in Fig. 4 for $k = 10$ time points, in Fig. 5 for $k = 20$ time points, in Fig. 6 for $k = 50$ time points, in Fig. 7 for $k = 100$ time points, in Fig. 8 for $k = 150$ time points, in Fig. 9 for $k = 200$ time points, respectively.

Table 5 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 100$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases p-value ≤ 0.05	58	32
The first type error rate estimate	0.058	0.032

Table 6 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 150$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases $p - \text{value} \leq 0.05$	61	25
The first type error rate estimate	0.061	0.025

Table 7 Point estimates of the first type error rates for the log-rank test and the proposed method for $k = 200$ time points

	Method	
	The log-rank test	The proposed method
# of simulated cases in total	1000	1000
# of cases p-value ≤ 0.05	60	21
The first type error rate estimate	0.060	0.021

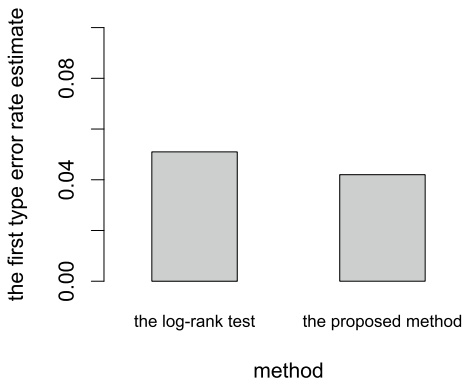


Fig. 4 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 10$ time points

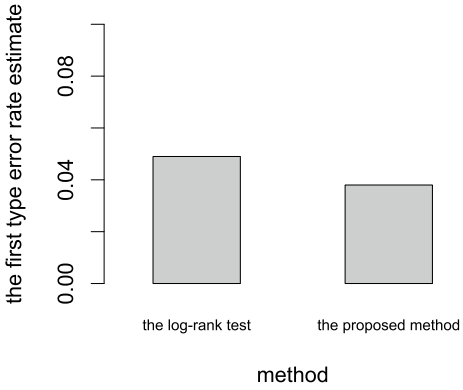


Fig. 5 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 20$ time points

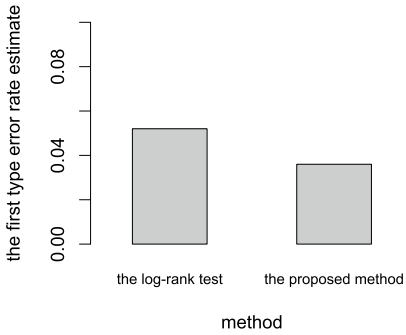


Fig. 6 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 50$ time points

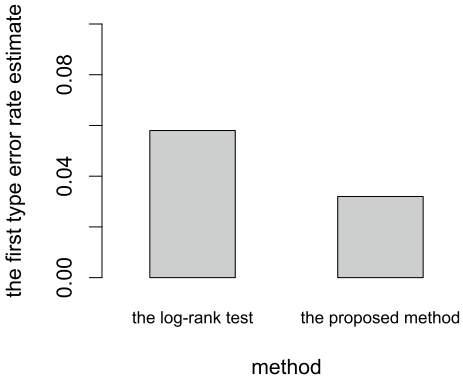


Fig. 7 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 100$ time points

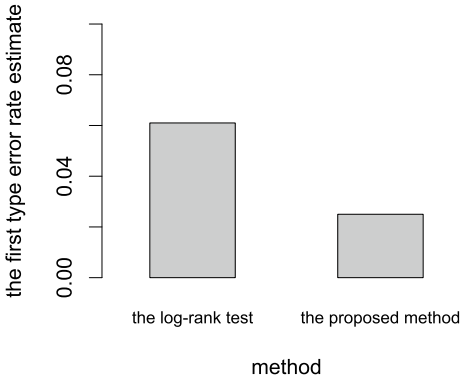


Fig. 8 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 150$ time points

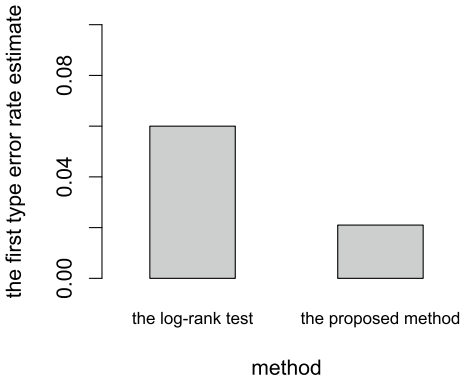


Fig. 9 A barplot comparing the first type error rate for the log-rank test and the proposed method for $k = 200$ time points

Whereas the log-rank test output point estimates of the first type error rate about 0.05 or even higher, the method introduced above returned a point estimate of the first type error rate about 0.04–0.03, therefore, in general, lower than the one for the log-rank test as demonstrated in Tables 2, 3, 4, 5, 6, 7 and Figs. 4, 5, 6, 7, 8, 9.

Thus, the proposed method seems to be more robust than the log-rank test, based on the simulation described above. The first type of error setting follows the alpha level's common value equal to 0.05, as usual in applied sciences.

5 Discussion

The introduced alternative to the log-rank test seem to return lower values of the first type errors, which was proven by multiple simulations following different scenarios for various numbers of considered time points.

In each of the scenarios, the novel method outputs a lower first type error rate than the log-rank test. Furthermore, the log-rank test's first type error rates very likely demonstrated a slightly increasing trend—once the number of considered time points

increased, and, thus, the sample size also increased, the first error rate increased, too. This property is typical for classic inference statistical tests—the larger the sample used for inference test statistic computations, the higher chance of the first error as a result of the test.

It is also worth mentioning that there is likely another trend in decreasing the first type error rates of the novel method once the number of considered time points increases. This phenomenon requires other research; that could be related to the rates by which the numerator and denominator increase when numbers of orthogonal paths increase and increase sample sizes.

According to the authors' opinion, the main reason that the introduced method generally returns a lower first type error rate than the log-rank test is that the new method does not consider any pairs of two survival curves crossing each other. While the log-rank test works for crossing curves as they should not cross each other, the introduced method p-value, based on the formula (3), is generally increased by lowering the denominator with the subtraction of the crossing curves. Intuitively, generally higher p-values rarely could result in the test's false positive outcome, called the first type error. That could be one of the most important sources of the new method's high robustness.

High robustness, i.e. the low first type error rate, is crucial for all statistical inference tests where the false-positive result's risk is too hazardous. That is typical for researching new drug effects that are compared to golden standards or placebos – when a new drug would be falsely marked as better than a current golden standard by an inference test, consequently, such a drug could harm a patient because of lacking any (therapeutic) effects. This area could offer room for the introduced, highly robust method's application.

6 Conclusion

By calculating monotonic orthogonal paths in the grid of survival plot, we can get a ratio of the number of all pairs of the more distant paths to each other, which opposes the null hypothesis, and the number of all non-crossing pairs of possible paths. That is a suggested point estimate of the p-value of the proposed alternative to the classical log-rank test.

Based on the simulation, the introduced method proved to be of higher robustness than the log-rank test. So, the assumption-free version of the log-rank test seems to be a valid alternative for the comparison of two time-event curves since the proposed method generally reduces rates of the first type error. However, while the numerical estimation of the p-value seems to be a relatively simple-to-follow, exhaustive (greedy) calculation of exact values is of high asymptotic time complexity, and analytic derivations of the p-value formula require the following research. The analytic derivation of the novel method's p-value and a quick p-value analytic computation is a promising way on how to combine low asymptotic time complexity of the clas-

sic log-rank test (and therefore, high speed of calculation) and the robustness of the introduced method just in only one tool.

Besides, the method and the computational aspects could also be a topic for a new R package development.

Acknowledgements This research was supported by the grant ES410060 no. 18/2021 "A machine-learning and other non-conventional approaches to survival time-event predicting: an innovation of established methodology and application to rare events of various provenience" provided by IGA/A grant competition of University of Economics, Prague.

References

1. Kaplan, E.L., Meier, P.: Nonparametric estimation from incomplete observations. J. Am. Stat. Assoc. **53**(457–481), 1958 (1958). https://doi.org/10.1080/01621459.1958.10501452
2. Mantel, N.: Evaluation of survival data and two new rank order statistics arising in its consideration. Cancer Chemother. Rep. **3**, 163–170 (1966)
3. Li, H., Han, D., Hou, Y., Chen, H., Chen, Z.: Statistical inference methods for two crossing survival curves: a comparison of methods. PLOS ONE **10**, e0116774. https://doi.org/10.1371/journal.pone.0116774 (ed Chen, Z.)
4. Team, R.C.: A Language and Environment for Statistical Computing R Foundation for Statistical Computing. Vienna, Austria (2017)
5. Therneau T.: A Package for Survival Analysis in R. Vienna, Austria, R package version 3.1-12. https://CRAN.R-project.org/package=survival/
6. Kong, F.: Robust covariate-adjusted logrank tests. Biometrika **84**, 847–862 (1997). https://doi.org/10.1093/biomet/84.4.847
7. Song, R., Kosorok, M.R., Cai, J.: Robust covariate-adjusted log-rank statistics and corresponding sample size formula for recurrent events data. Biometrics **64**, 741–750 (2007). https://doi.org/10.1111/j.1541-0420.2007.00948.x
8. Peto, R., Peto, J.: Asymptotically efficient rank invariant test procedures. J. R. Stat. Soc. Ser. A (General) **135**, 185 (1972). https://doi.org/10.2307/2344317
9. Yang, S., Prentice, R.: Improved Logrank-type tests for survival data using adaptive weights. Biometrics **66**, 30–38 (2009). https://doi.org/10.1111/j.1541-0420.2009.01243.x
10. Li, C.: Doubly robust weighted log-rank tests and Renyi-type tests under non-random treatment assignment and dependent censoring. Stat. Methods Med. Res. **28**, 2649–2664 (2018). https://doi.org/10.1177/0962280218785926
11. Thomas, D.G.: Exact and asymptotic methods for the combination of 2×2 tables. Comput. Biomed. Res. **8**, 423–446 (1975). https://doi.org/10.1016/0010-4809(75)90048-8
12. Mehta, C.R., Patel, N.R., Gray, R.: Computing an exact confidence interval for the common odds ratio in several 2×2 contingency tables. J. Am. Stat. Assoc. **80**(969) (1985). https://doi.org/10.2307/2288562
13. Heinze, G., Gnant, M., Schemper, M.: Exact log-rank tests for unequal follow-up. Biometrics **59**, 1151–1157, 1985 (2003). https://doi.org/10.1111/j.0006-341x.2003.00132.x
14. Mosler, K.: Multivariate Dispersion, Central Regions, and Depth?: The Lift Zonoid Approach. Springer, New York (2002).ISBN: 0387954120
15. Smolinski, T.: Computational Intelligence in Biomedicine and Bioinformatics?: Current Trends and Applications. Springer, Berlin (2008).ISBN: 978-3-540-70776-9
16. Kulikov, A.: Combinatorial Pattern Matching?: 25th Annual Symposium, CPM 2014 Moscow, Russia, June 16–18, 2014, Proceedings. Springer, Cham (2014).ISBN: 978-3-319-07565-5
17. Štepánek, L., Habarta, F., Malá, I., Marek, L.: Analysis of asymptotic time complexity of an assumption-free alternative to the log-rank test. In: Proceedings of the 2020 Federated

Conference on Computer Science and Information Systems. IEEE (2020). https://doi.org/10.15439/2020f198
18. Štepánek, L., Habarta, F., Malá, I., Marek, L., Pazdrek, F.: A Machinelearning approach to survival time-event predicting: initial analyses using stomach cancer data. In: 2020 International Conference on e-Health and Bioengineering (EHB). IEEE (2020). https://doi.org/10.1109/ehb50910.2020.9280301
19. Štepánek, L., Kasal, P., Mešták, J.: Evaluation of facial attractiveness for purposes of plastic surgery using machine-learning methods and image analysis. In: 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom). IEEE (2018). https://doi.org/10.1109/healthcom.2018.8531195
20. Štepánek, L., Kasal, P., Mešták, J.: Machine-learning at the service of plastic surgery: a case study evaluating facial attractiveness and emotions using R language. In: Proceedings of the 2019 Federated Conference on Computer Science and Information Systems. IEEE (2019). https://doi.org/10.15439/2019f264
21. Štepánek, L., Kasal, P., Mešták, J.: Evaluation of facial attractiveness after undergoing rhinoplasty using tree-based and regression methods. In: 2019 E-Health and Bioengineering Conference (EHB). IEEE (2019). https://doi.org/10.1109/ehb47216.2019.8969932
22. Štepánek, L., Kasal, P., Mešták, J.: Machine-learning and R in plastic surgery—evaluation of facial attractiveness and classification of facial emotions in advances in intelligent systems and computing. Springer International Publishing 243–252 (2019). https://doi.org/10.1007/978-3-030-30604-5_22
23. Martinková, P., et al.: Semi-real-time analyses of item characteristics for medical school admission tests. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.) Proceedings of the 2017 Federated Conference on Computer Science and Information Systems, vol. 11, pp. 189–194. IEEE (2017). https://doi.org/10.15439/2017F380
24. Štepánek, L., Martinková, P.: Feasibility of computerized adaptive testing evaluated by Monte-Carlo and post-hoc simulations. In: Proceedings of the 2020 Federated Conference on Computer Science and Information Systems. IEEE (2020). https://doi.org/10.15439/2020f197

Zero Point Approach to Three-Dimensional Intuitionistic Fuzzy Transportation Problem

Velichka Traneva and Stoyan Tranev

Abstract The transportation problem (TP) is a special type of linear programming problem where the objective is to minimise the cost of distributing a product from a number of sources or origins to a number of destinations. In classical TPs the decision maker has information about the crisp values of the transportation costs, availability and demand of the products. Sometimes the parameters of TPs in real life are vague and fuzzy caused by uncontrollable market factors. Uncertain values can be represented by fuzzy sets (FSs) of Zadeh. The FSs have the degrees of membership and non-membership. The concept of intuitionistic fuzzy sets (IFSs) originated in 1983 as an extension of FSs. Atanasov's IFSs also have a degree of hesitansy to representing the uncertainty in the environment. In this paper we formulate a three-dimensional TP (3-D IFTP), in which the transportation costs, supply and demand values are intuitionistic fuzzy pairs (IFPs), depending on the diesel prices, road condition, weather and other factors. Additional constraints are included in the problem: limits for the transportation costs and the priorities of the requests will be defined. Its main objective is to determine the quantities of delivery from producers to buyers to maintain the supply and demand requirements at time (location, etc.) at the cheapest intuitionistic fuzzy transportation costs. The aim of the paper is to extend the fuzzy and intuitionistic fuzzy zero point method (IFZPM [46, 63]) to the three-dimensional IFZPM (3-D IFZPM) to find an optimal solution of the intuitionistic fuzzy TP (IFTP) using the intuitionistc fuzzy logic and index matrices (IMs) concepts, proposed by Atanassov. The solution algorithm is demonstrated by a numerical example.

Keywords Transportation problem · Intuitionistic fuzzy set · Fuzzy numbers

This work was supported by the Bulgarian Ministry of Education and Science under the National Research Program "Young scientists and postdoctoral students", approved by DCM # 577/ 17.08.2018.

V. Traneva (✉) · S. Tranev
"Prof. Asen Zlatarov" University, "Prof. Yakimov" Blvd, 8000 Bourgas, Bulgaria
e-mail: tranev@abv.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_16

1 Introduction

The TP originally proposed by Hitchcock in 1941 [27]. Dantzig, in 1951, used simplex method to the TP [16]. The first overall, finished method for solving TP ("method of potentials") is developed by Kantorovich in 1949 [32].

In classical TP the decision maker has information about the values of transportation cost, the demanded and offered quantities of the product. In real-life transportation problems, some of its parameters are uncertain due to climatic, road conditions or other market conditions. Fuzzy sets (FSs) concept is introduced by Zadeh in 1965 [67] to represent and handle imprecise data. Zadeh, in [67], is proposed the idea of a fuzzy set as a class of objects with unsharp boundaries, i.e. in which elements may belong to partial degrees, from the full belongingness to the full non-belongingness through all intermediate values. Let a set E be fixed. A **"fuzzy set" (IFS)** A in E is an object of the following form (see [67]):

$$A = \{\langle x, \mu_A(x)\rangle | x \in E\},$$

where $\mu_A : E \to [0, 1]$ define the degree of membership of the $x \in E$ and $0 \leq \mu_A(x) \leq 1$ for every $x \in E$.

The existing methods for solving fuzzy transportation problems (FTPs), depending upon the fuzziness of decision variables, can be divided into two groups [37]. In the first group, the decision variables are assumed as real numbers, a crisp decision is to be made to meet some decision criteria. In the second group, decision variables are assumed as fuzzy numbers.

1.1 A Brief Literature Review of the Methods for FTPs

In this section, a brief review of the basic research done in the area of single-objective fuzzy transportation problems, fuzzy transshipment problems and fuzzy solid transportation problems are presented.

Oheigeartaigh [45] is proposed an algorithm to find the crisp optimal solution of such fuzzy transportation problems in which the availabilities and demands are represented by triangular fuzzy numbers. Chanas et al., in 1984, is proposed a fuzzy linear programming model for solving TPs with clear transportation costs, fuzzy supply and demand values [14]. Chanas and Kuchta [15], in 1996, is proposed a method to find the crisp optimal solution of such fuzzy transportation problems in which all the cost parameters are represented by LR type fuzzy numbers. Gen et al. have given a genetic algorithm for finding an optimal solution of a bicriteria solid TP with fuzzy numbers (FNs) [24]. Jimenez and Verdegay, in 1999, researched fuzzy Solid TP with trapezoidal FNs and presented a genetic approach for solving FTP [30]. Kikuchi [38] is represented all the parameters of FTPs by triangular fuzzy numbers and used the fuzzy linear programming approach to find the set of values such that the smallest

membership grade among them is maximized. Ahlatcioglu et al. [1] proposed an algorithm for finding the crisp optimal solution of fuzzy transportation problem by converting all the fuzzy parameters into intervals. Saad and Abbas [51] discussed the solution algorithm for solving the transportation problem in fuzzy environment. Liu and Kao [42] demostrated a method, based on Zadeh's extension principle, to find the optimal solution of the trapezoidal FTPs. Liang et al. [41] is described an interactive possibilistic linear programming approach for finding the crisp optimal solution of transportation problems with imprecise cost, availabilities and demand. Gani and Samuel [22] are proposed an algorithm for finding the fuzzy optimal solution of such fuzzy transportation problems in which availabilities and demands are represented by triangular fuzzy numbers. Liu and Kao [43] are extended the existing method proposed in their paper [42] to find a fuzzy optgal solution of such fuzzy solid transportation problems in which all the parameters are represented by trapezoidal fuzzy numbers. Smimou [56] is proposed a method to find the crisp optimal solution of a fuzzy transshipment model with fuzzy costs and transshipment model with fuzzy supply and fuzzy demand. Dinagar and Palanivel [20] are described fuzzy Vogel's approximation method and modified distribution method for determining an initial solution of trapezoidal FTPs. Das and Baruah [17] are proposed the use of Vogel's approximation method to find a fuzzy initial basic feasible solution and a modified method to find the optimal solution of such FTPs in which all the parameters are represented by triangular fuzzy numbers. Pandian and Natarajan, in 2010, studied Zero point method for solution for FTP with trapezoidal fuzzy parameters [46]. Improved zero point methods were described in (see [33, 47, 52, 53]) for solving trapezoidal and triangular FTP. De and Yadav [18] are modified the method due to Kikuchi [38] by using the trapezoidal fuzzy numbers instead of the triangular fuzzy numbers. Guzel [26] proposed a method to find the crisp optimal solution of FTPs with triangular fuzzy parameters. Kaur and Kumar, in 2012, introduced fuzzy least cost method, fuzzy north west corner rule and fuzzy Vogel approximation method for determining of an optimal solution of FTP [36]. Basirzadeh [13] has found a fuzzy optimal solution of fully FTPs by transforming the fuzzy parameters into the crisp parameters using classical algorithms. Venkatachalapathy [54] proposed a new method, a modified Vogel's approximation method, for finding a fuzzy optimal solution of fully fuzzy transportation problems. Gani et al. [21] used Arsham and Khan's simplex algorithm [3] to find a fuzzy optimal solution of FTPs with trapezoidal fuzzy parameters. A comparative analysis on the FTPs [44] was made and the conclusion has given that the zero point method is better than both the modified distribution method and Vogel's Approximation method. Patil and Chandgude, in 2012, performed "Fuzzy Hungarian approach" for TP with trapezoidal FNs [48]. Jahihussain and Jayaraman, in 2013, presented a zero suffix method for obtaining an optimal solution for FTPs with triangular and trapezoidal FNs (see [28, 29]). Zero suffix method to solve FTP after its converting into the crisp problem was applied in [28, 57]. A fuzzified version of zero suffix method was performed and applied in [49], in 2018, to FTPs. Shanmugasundari and Ganesan, in 2013, proposed a fuzzy modified distribution algorithm and a fuzzy approximation method of Vogel to solve FTP with FNs [55]. Fully FTPs is resolved in [19], in 2017, using a new method, based on

the Hungarian and MODI algorithm. Two new methods for finding a fuzzy optimal solution of TPs with the LR flat fuzzy numbers are proposed by Kaur, Kacprzyk and Kumar [37], based on the tabular representation and on the fuzzy linear programming formulation.

In 1983, Atanassov proposed the IFSs [5], which is an extension of FSs of Zadeh. IFSs are a more flexible tool for reflecting ambiguity in parameters than fuzzy ones because they have a hesitancy degree.

1.2 A Brief Literature Review of the Methods for IFTPs

Aggarwal and Gupta, in 2013, described an procedure for solving intuitionistic fuzzy TP (IFTP) with trapezoidal IFNs via ranking method [25]. In 2012 Gani et al. [22] introduced the solution of IFTP by using zero suffix algorithms. Gani and Abbas, in 2014 [23], and Kathirvel, and Balamurugun, in 2012 (see [34, 35]), proposed a method for solving TP in which the quantities demanded and offered are represented in the form of the trapezoidal intuitionistic FNs (IFNs). Antony et al. used Vogel's approximation method for solving triangular IFTP in 2014 [2]. "PSK method" for finding an optimal solution to IFTPs was presented by Kumar and Hussain in 2015 [39]. In [64], we have proposed for the first time the intuitionistic fuzzy Zero suffix method (IFZSM) to determine an optimal solution of the IFTP, interpreted by the IFSs and IMs [6] concepts.

In [31] is proposed the solution of the transportation problem (TP) in the form of intuitionistic fuzzy logic by using Zero point maximum allocation method. In [63], we have proposed for the first time intuitionistic fuzzy zero point method (IFZPM) to solve optimally a type of TP, in which the transportation costs, supply and demand quantities are IFPs. The constraints are formulated to the problem additionally: limits to the transportation costs. The zero point method for finding an optimal solution for IFTPs is free from the problem of degeneracy. In this paper, we extend the intuitionistic fuzzy Zero point method (IFZPM [63]) to the three-dimensional (3-D IFZPM) depending by a time-moment or a location using the intuitionistc fuzzy logic and index matrices (IMs) concepts, proposed by Atanassov. Additional constraints are included in the problem: limits for the transportation costs and the priorities of the requests will be defined.

The advantages of the algorithm are that it can be easy generalized for an application to multidimensional data and can be applied to both the TP with clear or known parameters, and with intuitionistic fuzzy ones. The structure of this paper is as follows: Sect. 2 of the research contains the basic concept of the theories of the IMs and the IFPs. In Sect. 3, we formulate 3-D IFTP and propose Zero point algorithm for its solution extending the fuzzy and intuitionistic fuzzy zero point method [46, 63] and using the concepts of IMs and IFSs. The reliability of the proposed approach is demonstrated by an example in Sect. 4. Section 5 outlines the conclusion and some directions for future research.

2 Preliminaries

In this section we recall some basic definitions on intuitionistic fuzzy pairs from (see [7, 9, 11, 58, 65]) and on index matrix apparatus from (see [8, 61]).

2.1 Short Remarks on Intuitionistic Fuzzy (IF) Logic

The **IFP** has the form of an ordered pair $\langle a, b\rangle = \langle \mu(p), \nu(p)\rangle$, where $a, b \in [0, 1]$ and $a + b \leq 1$, that is used as an evaluation of a proposition p (see [9, 11]). $\mu(p)$ and $\nu(p)$ respectively determine the "truth degree" (degree of membership) and "falsity degree" (degree of non-membership).

Let us have two IFPs $x = \langle a, b\rangle$ and $y = \langle c, d\rangle$. We recall some basic operations [7, 11, 12, 18, 50, 58]:

$$\begin{array}{cc}
\neg x = \langle b, a\rangle; & x \wedge_1 y = \langle \min(a, c), \max(b, d)\rangle; \\
x \vee_1 y = \langle \max(a, c), \min(b, d)\rangle; & x \wedge_2 y = x + y = \langle a + c - a.c, b.d\rangle; \\
x @ y = \langle \frac{a+c}{2}, \frac{b+d}{2}\rangle; & x \vee_2 y = x.y = \langle a.c, b + d - b.d\rangle; \\
\alpha.x = \langle 1 - (1 - a)^{\alpha}, b^{\alpha}\rangle & (\text{for } \alpha = n \text{ or } 1/n \ (n \in N));
\end{array}$$

$$x - y = \langle \max(0, a - c), \min(1, b + d, 1 - a + c)\rangle;$$

and relations with IFPs

$$\begin{array}{llll}
x \geq y \ \text{ iff } \ a \geq c \text{ and } b \leq d; & x \geq_{\Box} y \ \text{ iff } \ a \geq c; \\
x \geq_{\diamond} y \ \text{ iff } \ b \leq d; & x \geq_R y \ \text{ iff } \ R_{\langle a,b\rangle} \leq R_{\langle c,d\rangle};
\end{array} \tag{1}$$

where $R_{\langle a,b\rangle} = 0.5(2 - a - b)(1 - a)$ [58].

The IFP x is an **"intuitionistic fuzzy false pair" (IFFP)** if and only if $a \leq b$, while x is a **"false pair" (FP)** iff $a = 0, b = 1$.

Let a set E be fixed. An **"intuitionistic fuzzy set" (IFS)** A in E is an object of the following form (see [7]):

$$A = \{\langle x, \mu_A(x), \nu_A(x)\rangle | x \in E\},$$

where $\mu_A : E \to [0, 1]$ and $\nu_A : E \to [0, 1]$ define the degrees of membership and non-membership of the $x \in E$, respectively, and $0 \leq \mu_A(x) + \nu_A(x) \leq 1$ for every $x \in E$.

2.2 Definition, Operations and Relations over 3-D Intuitionistic Fuzzy Index Matrices

Let $\mathscr{I}$ be a fixed set. By three-dimensional intuitionistic fuzzy index matrix (3-D IFIM) $[K, L, H, \{\langle \mu_{k_i,l_j,h_g}, \nu_{k_i,l_j,h_g}\rangle\}]$ with index sets K, L and H ($K, L, H \subset \mathscr{I}$), we denote the object:

$$\equiv \begin{array}{c|cccccc} h_g \in H & l_1 & \dots & l_j & \dots & l_n \\ \hline k_1 & \langle \mu_{k_1,l_1,h_g}, \nu_{k_1,l_1,h_g}\rangle & \dots & \langle \mu_{k_1,l_j,h_g}, \nu_{k_1,l_j,h_g}\rangle & \dots & \langle \mu_{k_1,l_n,h_g}, \nu_{k_1,l_n.h_g}\rangle \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ k_m & \langle \mu_{k_m,l_1,h_g}, \nu_{k_m,l_1,h_g}\rangle & \dots & \langle \mu_{k_m,l_j,h_g}, \nu_{k_m,l_j,h_g}\rangle & \dots & \langle \mu_{k_m,l_n,h_g}, \nu_{k_m,l_n,h_g}\rangle \end{array},$$

where for every $1 \le i \le m$, $1 \le j \le n$, $1 \le g \le f$:

$$0 \le \mu_{k_i,l_j,h_g}, \nu_{k_i,l_j,h_g}, \mu_{k_i,l_j,h_g} + \nu_{k_i,l_j,h_g} \le 1.$$

Following [8, 59], we recall some operations over two IMs $A = [K, L, H, \{\langle \mu_{k_i,l_j,h_g}, \nu_{k_i,l_j,h_g}\rangle\}]$ and $B = [P, Q, E, \{\langle \rho_{p_r,q_s,e_d}, \sigma_{p_r,q_s,e_d}\rangle\}]$:
Negation $\neg A = [K, L, H, \{\langle \nu_{k_i,l_j,h_g}, \mu_{k_i,l_j,h_g}\rangle\}]$.
Addition-$(\circ, *)$

$$A \oplus_{(\circ,*)} B = [K \cup P, L \cup Q, H \cup E, \{\langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y}\rangle\}], \text{ where}$$

$$\langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y}\rangle$$

$$= \begin{cases} \langle \mu_{k_i,l_j,h_g}, \nu_{k_i,l_j,h_g}\rangle, & \text{if } t_u = k_i \in K, v_w = l_j, x_y = h_g \in H - E \\ & \text{or } t_u = k_i \in K, v_w = l_j \in L - Q, x_y = h_g \in H \\ & \text{or } t_u = k_i \in K - P, v_w = l_j \in L, x_y = h_g \in H \\ \\ \langle \rho_{p_r,q_s,e_d}, \sigma_{p_r,q_s,e_d}\rangle, & \text{if } t_u = p_r \in P, v_w = q_s \in Q, x_y = e_d \in E - H \\ & \text{or } t_u = p_r \in P, v_w = q_s \in Q - L, x_y = e_d \in E \\ & \text{or } t_u = p_r \in P - K, v_w = q_s \in Q, x_y = e_d \in E \\ \\ \langle \circ(\mu_{k_i,l_j,h_g}, \rho_{p_r,q_s,e_d}), & \text{if } t_u = k_i = p_r \in K \cap P, v_w = l_j = q_s \in L \cap Q \\ *(\nu_{k_i,l_j,h_g}, \sigma_{p_r,q_s,e_d})\rangle, & \text{and } x_y = h_g = e_d \in H \cap E \\ \\ \langle 0, 1\rangle, & \text{otherwise} \end{cases}$$

where $\langle \circ, *\rangle \in \{\langle \max, \min\rangle, \langle \min, \max\rangle, \langle \text{ average, average}\rangle\}$.
Termwise subtraction-(max, min):
$A -_{(\max,\min)} B = A \oplus_{(\max,\min)} \neg B$. **Termwise multiplication**-(min, max)

$$A \otimes_{(\min,\max)} B = [K \cap P, L \cap Q, H \cap R, \{\langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y}\rangle\}],$$

where $\langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y} \rangle = \langle \min(\mu_{k_i,l_j,h_g}, \rho_{p_r,q_s,e_d}), \max(\nu_{k_i,l_j,h_g}, \sigma_{p_r,q_s,e_d}) \rangle$.

The definition of the operations $A \otimes_{(\min,\min)} B$ and $A \otimes_{(\min,\max)} B$ are similar.
Transposition: A' is the transposed IM of A.
Reduction: We use symbol "$\perp$" for lack of some component in the separate definitions. In some cases, it is suitable to change this symbol with "0". The operation $(k, \perp, \perp)$-reduction of a given IM A is defined by:

$$A_{(k,\perp,\perp)} = [K - \{k\}, L, H, \langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y} \rangle], \text{ where}$$

$\langle \phi_{t_u,v_w,x_y}, \psi_{t_u,v_w,x_y} \rangle = \langle \mu_{k_i,l_j,h_g}, \nu_{k_i,l_j,h_g} \rangle$ for $t_u = k_i \in K - \{k\}$, $v_w = l_j \in L$ and $x_y = h_g \in H$. The definitions of the $(\perp, l, \perp)$- and $(\perp, \perp, h)$-reductions are analogous.
Projection: Let $M \subseteq K$, $N \subseteq L$ and $U \subseteq H$. Then, $pr_{M,N,U} A = [M, N, U, \{b_{k_i,l_j,h_g}\}]$, where for each $k_i \in M$, $l_j \in N$ and $h_g \in U$, $b_{k_i,l_j,h_g} = a_{k_i,l_j,h_g}$.
Substitution: Let the IM $A = [K, L, H, \{a_{k,l,h}\}]$ be given. Local substitution over A is defined for the pair of indices (p, k) by

$$\left[\frac{p}{k}; \perp; \perp\right] A = \left[(K - \{k\}) \cup \{p\}, L, H, \{a_{k,l,h}\}\right].$$

Index type operations:

$$AGIndex_{\{(\min/\max)/(\min_\square/\max_\square)/(\min_\diamond/\max_\diamond)(\min_R/\max_R)\}(\not\perp)}(A) = \langle k_i, l_j, h_g \rangle$$

finds the index of the minimum/ maximum element of A with no empty value in accordance with the relations (1).

$$AGIndex_{\{(\min/\max)/(\min_\square/\max_\square)/(\min_\diamond/\max_\diamond)(\min_R/\max_R)\}(\not\perp)(\notin F)}(A) = \langle k_i, l_j, h_g \rangle$$

presents the index of the minimum/ maximum element between the elements of A, whose indexes $\notin F$, with no empty value in accordance with the relations (1).

$$Index_{\{(\min/\max)/(\min_\square/\max_\square)/(\min_\diamond/\max_\diamond)(\min_R/\max_R)\}(\not\perp),k_i,h_g}(A)$$
$$= \{\langle k_i, l_{v_1}, h_g \rangle, \ldots, \langle k_i, l_{v_x}, h_g \rangle, \ldots, \langle k_i, l_{v_V}, h_g \rangle\},$$

where $\langle k_i, l_{v_x}, h_g \rangle$ (for $1 \le i \le m, 1 \le v \le n, 1 \le x \le V, 1 \le g \le f$) are the indices of the minimum/ maximum IFFP of the k_i-th index of the dimension K of A for a fixed h_g with no empty value in accordance with the relations (1).

$$Index_{(\not\perp)/(1)}(A) = \{\langle k_1, l_{v_1}, h_g \rangle, \ldots, \langle k_i, l_{v_i}, h_g \rangle, \ldots, \langle k_m, l_{v_m}, h_g \rangle\},$$

where $\langle k_i, l_{v_i}, h_g \rangle$ (for $1 \le i \le m, 1 \le g \le f$) are the indices of the elements of A, whose cell are full or equal to 1.

$$Index_{(\max \mu(\nu)),k_i,h_g}(A) = \{\langle k_i, l_{v_1}, h_g\rangle, \ldots, \langle k_i, l_{v_x}, h_g\rangle, \ldots, \langle k_i, l_{v_V,h_g}\rangle\},$$

where $\langle k_i, l_{v_x}, h_g\rangle$ (for $1 \le i \le V$, $1 \le x \le n$, $1 \le g \le f$) are the indices of the IFFP of the k_i-th row of A for a fixed h_g, for which $\mu(\nu)_{k_i,l_{v_x},h_g}$ is maximum.

$$Index_{(\max \mu(\nu)),l_j,h_g}(A) = \{\langle k_{w_1}, l_j, h_g\rangle, \ldots, \langle k_{w_y}, l_j, h_g\rangle, \ldots, \langle k_{w_W}, l_j, h_g\rangle\},$$

where $\langle k_{w_y}, l_j, h_g\rangle$ (for $1 \le y \le W$, $1 \le j \le n$, $1 \le g \le f$) are the indices of the IFFP of l_j-th column of A for a fixed h_g, for which $\mu(\nu)_{k_{w_y},l_j,h_g}$ is maximum.

Aggregation operations

Let us use the operations $\#_q$, $(q \le i \le 3)$ from [66] for scaling aggregation operations over two IFPs $x = \langle a, b\rangle$ and $y = \langle c, d\rangle$:

$x\#_1 y = \langle min(a, c), max(b, d)\rangle$;

$x\#_2 y = \langle average(a, c), average(b, d)\rangle$;

$x\#_3 y = \langle max(a, c), min(b, d)\rangle$.

The following inequality holds: $x\#_1 y \le x\#_2 y \le x\#_3 y$66.

Let $k_0 \notin K$ and $h_0 \notin H$ be fixed indexes. The definition of the aggregation operation for an IM A by dimensions K and H are according to [8, 62, 66]:

$$\alpha_{K,\#_q}(A, k_0) = \begin{array}{c|ccc} h_g & l_1 & \ldots & l_n \\ \hline k_0 & \mathop{\#_q}\limits_{i=1}^{m} \langle \mu_{k_i,l_1',h_g}, \nu_{k_i,l_1,h_g}\rangle & \ldots & \mathop{\#_q}\limits_{i=1}^{m} \langle \mu_{k_i,l_n,h_g}, \nu_{k_i,l_n,h_g}\rangle \end{array},$$

where $1 \le q \le 3$.

$$\alpha_{H,\#_s^*}(A, h_0) = \begin{array}{c|ccc} h_0 & l_1 & \ldots & l_n \\ \hline k_1 & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_1,l_1,h_g}, \nu_{k_1,l_1,h_g}\rangle & \ldots & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_1,l_n,h_g}, \nu_{k_1,l_n,h_g}\rangle \\ \vdots & \vdots & \ldots & \vdots \\ k_i & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_i,l_1,h_g}, \nu_{k_i,l_1,h_g}\rangle & \ldots & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_i,l_n,h_g}, \nu_{k_i,l_n,h_g}\rangle \\ \vdots & \vdots & \ldots & \vdots \\ k_m & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_m,l_1,h_g}, \nu_{k_m,l_1,h_g}\rangle & \ldots & \mathop{\#_s^*}\limits_{g=1}^{f} \langle \mu_{k_m,l_n,h_g}, \nu_{k_m,l_n,h_g}\rangle \end{array},$$

where $1 \le q \le 3$.

Aggregate global internal operation: $AGIO_{\oplus_{(\max,\min)}}(A)$.

This operation finds the addition of all elements of A.

Internal subtraction of the components of the IM A [60, 61, 65]:

$$IO_{-(\min,\max)}(\langle k_i, l_j, h_g, A\rangle, \langle p_r, q_s, e_d, B\rangle) = [K, L, H, \{\langle \gamma_{t_u,v_w,z_y}, \delta_{t_u,v_w,z_y}\rangle\}],$$

where $k_i \in K,\ l_j \in L,\ h_g \in H;\ p_r \in P,\ q_s \in Q, e_d \in E$ and

$$\langle \gamma_{t_u,v_w,z_y}, \delta_{t_u,v_w,z_y} \rangle$$

$$= \begin{cases} \langle \mu_{t_u,v_w,z_y}, \nu_{t_u,v_w,z_y} \rangle, & \text{if } t_u \neq k_i \in K, v_w \neq l_j \in L, \\ & z_y \neq h_g \in H \\ \langle \max(0, \mu_{k_i,l_j,h_g} - \rho_{p_r,q_s,e_d}), & \text{if } t_u = k_i \in K, \\ \min(1, \nu_{k_i,l_j,h_g} + \sigma_{p_r,q_s,e_d}, 1 - \mu_{k_i,l_j,h_g} + \rho_{p_r,q_s,e_d}) \rangle, & v_w = l_j \in L, z_y = h_g \in H. \end{cases}$$

The non-strict relation "inclusion about value" The form of this type of relations between two IMs A and B is as follows:

$$A \subseteq_v B \text{ iff } (K = P)\&(L = Q)\&(H = R)\&(\forall k \in K)(\forall l \in L)(\forall h \in H)$$

$$(a_{k,l,h} \leq b_{k,l,h}).$$

3 Zero Point Approach to the 3-D IFTP

Let us extend the IFTP from [63] to the 3-D IFTP with restrictions: A trader supplies a product to n different companies (consumers) $\{l_1, \ldots, l_j, \ldots, l_n\}$ in a time-moment $h_g (g = 1, \ldots, f)$ after delivery of that product from different m manifacturers (producers) $\{k_1, \ldots, k_i, \ldots, k_m\}$ in quantities c_{k_i,R,h_g} (for $i = 1, \ldots, m$; $g = 1, \ldots, f$). H is a fixed time scale and h_g is its element. The interpretation of H may also be different such as location or other. Let the consumers (destinations) in a time-moment h_g (from a location h_g) need this product in quantities of c_{Q,l_j,h_g} (for $i = 1, \ldots, m$; $g = 1, \ldots, f$). Let c_{k_i,l_j,h_g} be the intuitionistic fuzzy cost for transporting one unit quantity of the product from the k_i-th producer to the l_j-th consumer at time h_g; x_{k_i,l_j,h_g}—the number of units of the product, transported from the k_i-th source to the l_j-th destination at time h_g and IFPs c_{pl,l_j,h_g} (for $j = 1, \ldots, n$; $g = 1, \ldots, f$) are limits for the transportation costs of a delivery a product from the k_i-th manifacturer to the l_j-th destination at h_g. $c_{k_i,l_j,T}$—the quantity of units of the product transported by the k_i-th producer to the l_j-th consumer for the defined time period. The priorities of consumer requests will be defined in advance.

All parameters, involved in the problem, are IFPs. For estimating the transportation costs in the form of IFPs, we can use the expert approach described in detail in [7]. Each expert needs to evaluate at least a part of the alternatives in terms of their performance with respect to each defined criterion. The experts is not sure about the transportation costs due the climatic and traffic conditions, or economic factors. He hesitates in prediction of the transportation cost due to changes in some uncontrollable factors. The transportation costs are evaluated as intuitionistic fuzzy numbers after a

thorough discussion, interpreted by the intuitionistic fuzzy concept: these numbers express a "positive" and a "negative" evaluations, respectively. The reliability of the expert assessment (confidence in her/his evaluation with respect to each criterion) may be involved in the evaluation process.

The purpose of the trader is how to satisfy the requests of the users so that the intuitionistic fuzzy transportation cost is minimum for the entire period of time according to (1). Let us formulate the mathematical model of the above problem:

$$\begin{aligned}
&\text{An objective function: minimize } \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{g=1}^{f} c_{k_i,l_j,h_g} x_{k_i,l_j,h_g} \\
&\text{Subject to: } \sum_{j=1}^{n}\sum_{g=1}^{f} x_{k_i,l_j,h_g} = c_{k_i,R,h_g}, \qquad i = 1, 2, \ldots, m \\
&\text{Subject to: } \sum_{i=1}^{m}\sum_{g=1}^{f} x_{k_i,l_j,h_g} = c_{Q,l_j,h_g}, \qquad j = 1, 2, \ldots, n \\
&\sum_{i=1}^{m}\sum_{j=1}^{n} x_{k_i,l_j,h_g} = c_{k_i,l_j,T}, \qquad g = 1, 2, \ldots, f
\end{aligned} \tag{2}$$

We add the constraints to the problem (2) as follows:

- c_{pl,l_j,h_g}, for $1 \le j \le n$, $1 \le g \le f$—an intuitionistic fuzzy upper limit to the corresponding transportation cost of delivery a particular product from the k_i-th source to the l_j-th destination;
- The priorities of consumer requests will be defined in advance in IM $PREF$.

Note: The operations "addition" and "multiplication", used in the problem (2) are those for IFPs, defined in Sect. 2.

The transportation costs of the problem (2) for delivery from a given manifacturer to a given user are entered in the cost IM C:

$$C[K, L, H] =$$

$h_g \in H$	l_1	$\ldots$	l_n	R	pu
k_1	$\langle \mu_{k_1,l_1,h_g}, \nu_{k_1,l_1,h_g} \rangle$	$\ldots$	$\langle \mu_{k_1,l_n,h_g}, \nu_{k_1,l_n,h_g} \rangle$	$\langle \mu_{k_1,R,h_g}, \nu_{k_1,R,h_g} \rangle$	$\langle \mu_{k_1,pu,h_g}, \nu_{k_1,pu,h_g} \rangle$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
k_m	$\langle \mu_{k_m,l_1,h_g}, \nu_{k_m,l_1,h_g} \rangle$	$\ldots$	$\langle \mu_{k_m,l_n,h_g}, \nu_{k_m,l_n,h_g} \rangle$	$\langle \mu_{k_m,R,h_g}, \nu_{k_m,R,h_g} \rangle$	$\langle \mu_{k_m,pu,h_g}, \nu_{k_m,pu,h_g} \rangle$
Q	$\langle \mu_{Q,l_1,h_g}, \nu_{Q,l_1,h_g} \rangle$	$\ldots$	$\langle \mu_{Q,l_n,h_g}, \nu_{Q,l_n,h_g} \rangle$	$\langle \mu_{Q,R,h_g}, \nu_{Q,R,h_g} \rangle$	$\langle \mu_{Q,pu,h_g}, \nu_{Q,pu,h_g} \rangle$
pl	$\langle \mu_{pl,l_1,h_g}, \nu_{pl,l_1,h_g} \rangle$	$\ldots$	$\langle \mu_{pl,l_n,h_g}, \nu_{pl,l_n,h_g} \rangle$	$\langle \mu_{pl,R,h_g}, \nu_{pl,R,h_g} \rangle$	$\langle \mu_{pl,pu,h_g}, \nu_{pl,pu,h_g} \rangle$
pu_1	$\langle \mu_{pu_1,l_1,h_g}, \nu_{pu_1,l_1,h_g} \rangle$	$\ldots$	$\langle \mu_{pu_1,l_n,h_g}, \nu_{pu_1,l_n,h_g} \rangle$	$\langle \mu_{pu_1,R,h_g}, \nu_{pu_1,R,h_g} \rangle$	$\langle \mu_{pu_1,pu,h_g}, \nu_{pu_1,pu,h_g} \rangle$

,

where $K = \{k_1, k_2, \ldots, k_m, Q, pl, pu_1\}$, $L = \{l_1, l_2, \ldots, l_n, R, pu\}$, $H = \{h_1, h_2, \ldots, h_f, T\}$ and for $1 \le i \le m$, $1 \le j \le n$, $1 \le g \le f$, $\{c_{k_i,l_j,h_g}, c_{k_i,R,h_g}, c_{k_i,pu,h_g}, c_{pl,l_j,h_g}, c_{pl,R,h_g}, c_{pl,pu,h_g}, c_{Q,l_j,h_g}, c_{Q,R,h_g}, c_{Q,pu,h_g}, c_{pu_1,l_j,h_g}, c_{pu_1,R,h_g}, c_{pu_1,pu,h_g}\}$ are IFPs.

Let we denote by $|K| = m + 3$ the number of elements of the set K; then $|L| = n + 2$ and $|H| = f + 1$. We also define the IM

$$X[K*, L*, H*, \{x_{k_i,l_j,h_g}\}] = \begin{array}{c|ccccc} h_g \in H* & l_1 & \dots & l_j & \dots & l_n \\ \hline k_1 & x_{k_1,l_1,h_g} & \dots & x_{k_1,l_j,h_g} & \dots & x_{k_1,l_n,h_g} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ k_m & x_{k_m,l_1,h_g} & \dots & x_{k_m,l_j,h_g} & \dots & x_{k_m,l_n,h_g} \end{array},$$

where $K*=\{k_1, k_2, \dots, k_m\}$, $L*=\{l_1, l_2, \dots, l_n\}$, $H*=\{h_1, h_2, \dots, h_f\}$, and for $1 \le i \le m$, $1 \le j \le n$, $1 \le g \le f$:

$$x_{k_i,l_j,h_g} = \langle \rho_{k_i,l_j,h_g}, \sigma_{k_i,l_j,h_g} \rangle$$

Then, let us we create the following auxiliary index matrices:

(1) $S = [K, L, H\{s_{k_i,l_j,h_g}\}]$, such that $S = C$ i.e. ($s_{k_i,l_j,h_g} = c_{k_i,l_j,h_g}$ $\forall k_i \in K, \forall l_j \in L, \forall h_g \in H$);

(2)

$$D[K*, L*, H*] = \begin{array}{c|ccccc} h_g \in H* & l_1 & \dots & l_j & \dots & l_n \\ \hline k_1 & d_{k_1,l_1,h_g} & \cdots & d_{k_1,l_j,h_g} & \cdots & d_{k_1,l_n,h_g} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ k_m & d_{k_m,l_1,h_g} & \dots & d_{k_m,l_j,h_g} & \dots & d_{k_m,l_n,h_g} \end{array},$$

where for $i = 1, \dots, m$; $j = 1, \dots, n$; $g = 1, \dots, f$: $d_{k_i,l_j,h_g} = \{1 \text{ or } 2\}$ depending on whether the elements s_{k_i,l_j,h_g} of S are crossed out with 1 or 2 lines.

(3)

$$RC[K*, e_0, h_g] = \begin{array}{c|c} h_g & e_0 \\ \hline k_1 & rc_{k_1,e_0,h_g} \\ \vdots & \vdots \\ k_m & rc_{k_m,e_0,h_g} \end{array},$$

where for $1 \le i \le m$: $rc_{k_i,l_j,h_g} = \{0 \text{ or } 1\}$ depending on whether the k_i-th index of $K*$ of the matrix S is crossed out at time h_g;

(4)

$$CC[r_0, L*, h_g] = \begin{array}{c|ccccc} h_g & l_1 & \dots & l_j & \dots & l_n \\ \hline r_0 & cc_{r_0,l_1,h_g} & \cdots & cc_{r_0,l_j,h_g} & \cdots & cc_{r_0,l_n,h_g} \end{array},$$

where for $1 \le j \le n$: $cc_{k_i,l_j,h_g} = \{0 \text{ or } 1\}$ depending on whether the l_j-th index of $L*$ of the matrix S is crossed out at time h_g;

(5) $RM[K*, pu, H*] = pr_{K*,pu,H*}C$ and $CM[pu_1, L*, H*] = pr_{pu_1,L*,H*}C$;

(6) $U[K*, L*, H*, \{u_{k_i,l_j,h_g}\}]$ and for $1 \le i \le m$, $1 \le j \le n$, $1 \le g \le f$:

$$u_{k_i,l_j,h_g} = \begin{cases} 0, & \text{if } c_{k_i,l_j,h_g} < c_{pl,l_j,h_g} \\ 1, & \text{otherwise} \end{cases};$$

(7) It is necessary to create the auxiliary IM of consumer priorities:

$$P[k_1, L*, H*, \{p_{k_1,l_j,h_g}\}],$$

where $L* = \{l_1, l_2, \ldots, l_n\}$, $H* = \{h_1, h_2, \ldots, h_f\}$ and $p_{k_1,l_j,h_g} = \langle \delta_{k_1,l_j,h_g}, \gamma_{k_1,l_j,h_g} \rangle$ is the priority of the l_j-th consumer at time h_g.

We compose the IM for the priorities of consumer requests

$$PREF[K*, L*, H*, \{pref_{k_i,l_j,h_g}\}]$$

$$= P \oplus_{(\max,\min)} \ldots \oplus_{(\max,\min)} \left[\frac{k_i}{k_1}; \perp; \perp\right] P \oplus_{(\max,\min)} \ldots \oplus_{(\max,\min)} \left[\frac{k_m}{k_1}; \perp; \perp\right] P$$

where $K* = \{k_1, k_2, \ldots, k_m\}$, $L* = \{l_1, l_2, \ldots, l_n\}$, $H* = \{h_1, h_2, \ldots, h_f\}$.

When starting the algorithm, $rc_{k_i,e_0,h_g} = cc_{r_0,l_j,h_g} = 0$, $u_{k_i,l_j,h_g} = 1$, $x_{k_i,l_j,h_g} = \langle 0, 1\rangle$, $c_{k_i,pu,h_g} = \langle \perp, \perp\rangle$, $c_{pu_1,L,h_g} = \langle \perp, \perp\rangle$ $(\forall k_i \in K*, \forall l_j \in L*, \forall h_g \in H*)$.

We will describe for the first time an extended zero point approach to the IFZPM from [63] for determining the optimal solution of the TP with intuitionistic fuzzy costs, demand and supply [63] using the concepts of IMs and IFPs. In the proposed program code is used a pseudocode and mathematical formulation.

Step 1. Let us create the 3-D IFIM C for the given problem and then, convert it into a balanced one ($\sum_{i=1}^{m} c_{k_i,R,h_g} = \sum_{j=1}^{n} c_{Q,l_j,h_g}$), if it is not.

The program executes the following operations:

– We define $3-D$ IMs as follows:

$S_1[Q, L*, h_g] = pr_{Q,L*,h_g}C$; $S_2[K*, R, h_g] = pr_{K*,R,h_g}C$ and let $\{k_{m+1}, l_{n+1}\} \notin K \cup L$.

We need to use the following IMs operations:

– If $\alpha_{K,\#_q}(S_1, l_{n+1}) \supset_v \left[\frac{Q}{R}; \perp\right](\alpha_{L,\#_q}(S_2, l_{n+1}))'$ (i.e. $\sum_{i=1}^{m} c_{k_i,R,h_g} > \sum_{j=1}^{n} c_{Q,l_j,h_g}$), then *introduce dummy column l_{n+1} having all its costs as $\langle 0, 1\rangle$ and execute operations for finding the demand at this dummy destination:*

$c_{Q,l_{n+1},h_g} = \sum_{i=1}^{m} c_{k_i,R,h_g} - \sum_{j=1}^{n} c_{Q,l_j,h_g}$; {Let us define $2-D$ IMs S_3, S_4, S_5 such that

$$S_3 = \alpha_{K,\#_q}(S_1, l_{n+1}) -_{(\max,\min))} \alpha_{L,\#_q}(\left[\frac{Q}{R}; \perp\right](S_2, l_{n+1}))';$$

$S_4 = [K*, l_{n+1}, h_g, \{\langle 0, 1\rangle\}]$; $S_5 = [K, l_{n+1}, h_g, \{c_{k_i,l_{n+1},h_g}\}] = S_3 \oplus_{(\max,\min)} S_4$.

The new matrix of costs is obtained by carrying out the operation "matrix addition": $C := C \oplus_{(\max,\min))} S_5$, go to *Step 2*.}

– If $\left[\perp; \frac{R}{Q}\right]\alpha_{K,\#_q}(S_1, k_{m+1}))' \subset_v \alpha_{L,\#_q}(S_2, k_{m+1}))'$ $\left(\text{i.e. } \sum_{i=1}^{m} c_{k_i,R,h_g} < \sum_{j=1}^{n} c_{Q,l_j,h_g}\right)$, then *introduce dummy row k_{m+1} having all its costs as $\langle 0, 1\rangle$ and execute operations for finding the demand at this dummy destination:* $c_{k_{m+1},R,h_g} = \sum_{i=1}^{m} c_{k_i,R,h_g} -$

$\sum_{j=1}^{n} c_{Q,l_j,h_g}$.
{Let us define $2-D$ IMs S_3, S_4, S_5 such that

$$S_3 = \alpha_{K,\#_q}(S_2, k_{m+1}) -_{(\max,\min))} \left[\perp; \frac{R}{Q}\right] \alpha_{L,\#_q}(S_1, k_{m+1}))';$$

$S_4[k_{m+1}, L*, h_g, \{\langle 0, 1\rangle\}]$; $S_5 = [k_{m+1}, L*, h_g, \{c_{k_{m+1},l_j,h_g}\}] = S_3 \oplus_{(\max,\min))} S_4$;
$C := C \oplus_{(\max,\min))} S_5$, go to *Step 2*. }
Step 2. Checking the conditions for limiting the transportation costs
for $g = 1$ to f
for $i = 1$ to m
for $j = 1$ to n

$$\{\text{If}\left(\left[\frac{k_i}{pl}; \perp\right] pr_{pl,l_j,h_g} C\right) \subseteq_v pr_{k_i,l_j,h_g} C, \text{ then } u_{k_i,l_j,h_g} = 1.\}$$

$EG = Index_{(1)}(U) = \{\langle k_{i_1}, l_{j_1}, h_{g_1}\rangle, \langle k_{i_2}, l_{j_2}, h_{g_2}\rangle, \ldots, \langle\langle k_{i_\phi}, l_{j_\phi}, h_{g_\phi}\rangle\}$;
For each $\langle k_i, l_j, h_g\rangle \in EG$, let us the element s_{k_i,l_j,h_g} of S is equal to $\langle 1, 0\rangle$ [40];
Let us create the IM DC, such that

$$DC := C_{(\{Q,pl,pu_1\},\{R,pu\},\{T\})} \otimes_{(\min,\max)} \neg PREF;$$

$C := pr_{K,R,H} C \oplus_{(\max,\min)} pr_{K,pu,H} C \oplus_{(\max,\min)} pr_{Q,L*,H} C$
$\oplus_{(\max,\min)} pr_{Q,R,T} C \oplus_{(\max,\min)} DC \oplus_{(\max,\min)} pr_{pl,L*,H} C \oplus_{(\max,\min)} pr_{pu_1,L*,H} C$.
Go to *Step 3*.
Step 3. Determination of zero membership value—$K*$ level For each index k_i of K at a fixed h_g $(g = 1, \ldots, f)$ of the matrix S, the smallest element is found in accordance with the relations (1) among the elements s_{k_i,l_j,h_g} $(j = 1, \ldots, n)$ and is saved to the right of the row, in the column pu. It is subtracted from all elements s_{k_i,l_j,h_g}, for $j = 1, 2, \ldots, n$. The code uses the operation $AGIO$ for finding the indexes of the minimum elements of the row:
for $g = 1$ to f
for $i = 1$ to m
for $j = 1$ to n
If $(c_{k_i,pu,h_g} = \langle\perp, \perp\rangle)$, then

$$\{AGIndex_{\{(\min)/(\min_\square)/(\min_\diamond)/(\min_R)\}}\left(pr_{k_i,L*,h_g} S\right) = \langle k_i, l_{v_j}, h_g\rangle;$$

If $u_{k_i,l_j,h_g} = 0$, then
$\{S_6[k_i, l_{v_j}, h_g] = pr_{k_i,l_{v_j},h_g} S$; $S_7 = \left[\perp; \frac{pu}{l_{v_j}}\right] S_6$; $S := S \oplus_{(\max,\min)} S_7.\}\}$
Aftere subtraction we get the matrix S with reduced prices.
for $g = 1$ to f
for $i = 1$ to m

for $j = 1$ to n
$\{IO_{-(\max,\min)}\left(\langle k_i, l_j, h_g, S\rangle, \langle k_i, pu, h_g, pr_{K*,pu,H*}S\rangle\right)\}$;
Go to *Step 4*.

Step 4. Determination of zero membership value—$L*$ level For each index l_j of L at a fixed h_g $(g = 1, \ldots, f)$ of the matrix S, the smallest element is found among the elements s_{k_i,l_j,h_g} $(i = 1, \ldots, m)$. It is saved at the bottom of the column, in line pu_1 and it is subtracted from all elements s_{k_i,l_j,h_g}, for $i = 1, 2, \ldots, m$.
for $g = 1$ to f
for $j = 1$ to n
for $i = 1$ to m
If $(c_{pu_1,l_j,h_g} = \langle \perp, \perp \rangle)$, then

$$\{AGIndex_{\{(\min)/(\min_\square)/(\min_\diamond)/(\min_R)\}}\left(pr_{K*,l_j,h_g}S\right) = \langle k_{w_i}, l_j, h_g\rangle;$$

$$\text{If } u_{k_i,l_j,h_g} = 0, \text{ then } \{ \text{ let us create two 3-D IMs } S_6 \text{ and } S_7 :$$

$$S_6[k_{w_i}, l_j, h_g] = pr_{k_{w_i},l_j,h_g}S;\ S_7 = \left[\frac{pu_1}{k_{w_i}}; \perp\right] S_6;\ S := S \oplus_{(\max,\min)} S_7; \}\}$$

We get the matrix S with reduced costs
for $j = 1$ to n
for $g = 1$ to f
for $i = 1$ to m
$\{IO_{-(\max,\min)}\left(\langle k_i, l_j, h_g, S\rangle, \langle pu_1, l_j, h_g, pr_{pu_1,L*,H*}S\rangle\right)\}$;
Go to *Step 5*.

Step 5. Determination of zero membership value—$H*$ level For each index k_i of K at a fixed l_j $(j = 1, \ldots, n)$ of the matrix S, the smallest element is found among the elements s_{k_i,l_j,h_g} $(g = 1, \ldots, f)$ and is recorded as the value of the element $s_{k_i,l_j,T}$. It is subtracted from all elements s_{k_i,l_j,h_g}, for $g = 1, 2, \ldots, f$ and go to *Step 6*.
for $j = 1$ to n
for $i = 1$ to m
for $g = 1$ to f
If $(c_{k_i,l_j,T} = \langle \perp, \perp \rangle)$, then

$$\{AGIndex_{\{(\min)/(\min_\square)/(\min_\diamond)/(\min_R)\}}\left(pr_{k_i,l_j,H*}S\right) = \langle k_i, l_j, h_{z_g}\rangle;$$

$$\text{If } u_{k_i,l_j,h_g} = 0, \text{ then } \{ \text{ let us create two 3-D IMs } S_6 \text{ and } S_7 :$$

$$S_6[k_i, l_j, h_{z_g}] = pr_{k_i,l_j,h_{z_g}}S;\ S_7 = \left[\perp; \perp; \frac{T}{h_{z_g}}\right] S_5;\ S := S \oplus_{(+)} S_6\}\}.$$

For each index k_i of K of S at a fixed l_j $(j = 1, \ldots, n)$ we subtract the smallest element $s_{k_i,l_j,h_{z_g}}$ from all elements s_{k_i,l_j,h_g} $(g = 1, \ldots, f)$.
We create IM $B = pr_{K*,l_j,T}S$.

for $i = 1$ to m
for $j = 1$ to n
for $g = 1$ to f
$\{IO_{-(\min,\max)}\left(\langle k_i, l_j, h_g, S\rangle, \langle k_i, l_j, T, B\rangle\right)\}$.

Step 6. Optimality criterion

(1) Check if each quantity of product offered at time h_g is less than or equal the total quantity of product required, whose reduced costs are with zero membership degrees.
for $g = 1$ to f
for $i = 1$ to m
$\{Index_{(\min\mu),k_i}(A) = \{\langle k_i, l_{v_1}, h_g\rangle, \ldots, \langle k_i, l_{v_x}, h_g\rangle, \ldots, \langle k_i, l_{v_V}, h_g\rangle;$
We create $3 - D$ IMs as follows:

$$G_{v_1}[k_i, l_{v_1}, h_g] = pr_{k_i,l_{v_1},h_g}C, \ldots, G_{v_V}[k_i, l_{v_V}, h_g] = pr_{k_i,l_{v_V},h_g}C,$$

and $G[k_i, R, h_g] = pr_{k_i,R,h_g}C$;

$$\text{If} \quad G[k_i, R, h_g] \subseteq_v G_{v_1} +_{(\max,\min)} \ldots +_{(\max,\min)} G_{v_x} + \ldots +_{(\max,\min)} G_{v_V},$$

then go to *Step 6.2*
else $\{RM[k_i, pu, h_g] = 1$ and go to *Step 7;*$\}$ $\}$

(2) Check if each required quantity at time h_g is less than or equal to the total offered quantity, whose reduced costs have zero membership degrees.
for $g = 1$ to f
for $j = 1$ to n
$Index_{(\min\mu),l_j,h_g}(A) = \{\langle k_{w_1}, l_j, h_g\rangle, \ldots, \langle k_{w_y}, l_j, h_g\rangle, \ldots, \langle k_{w_W}, l_j, h_g\rangle\}$;
We define $3 - D$ IMs as follows:

$$G_{w_1}[k_{w_1}, l_j, h_g] = pr_{k_{w_1},l_j,h_g}C, \ldots, G_{w_W}[k_{w_W}, l_j, h_g] = pr_{k_{w_W},l_j,h_g}C,$$

$$\text{and } G[Q, l_j, h_g] = pr_{Q,l_j,h_g}C;$$

$$\text{If} \quad G[Q, l_j, h_g] \subseteq_v G_{w_1} +_{(\max,\min)} \ldots +_{(\max,\min)} G_{w_y} + \ldots +_{(\max,\min)} G_{w_W},$$

then go to *Step 9*
else $\{CM[pu_1, l_j, h_g] = 1$ go to *Step 7.* $\}$ $\}$

Step 7. Revise the cost IM All elements $\langle 0, 1\rangle$ in the S are crossed out with minimum number of lines (horizontal, vertical or both). If there is no element $\langle 0, 1\rangle$ in a given row or column at time h_g (for $g = 1, \ldots, f$), then the element with the minimum degree of membership is crossed out from that row or column in the cost IM S obtained in *s*tep 6.

This step introduces IM $D[K*, L*, H*]$, which has the same dimensions as the X matrix. We use to mark whether an element in S is crossed out with a line to the direction of the dimension $K*$ or $L*$ at time h_g for $g = 1, \ldots, f$.

If $d_{k_i,l_j,h_g} = 1$ (for $1 \le i \le m$; $1 \le j \le n$; $1 \le g \le f$), then s_{k_i,l_j,h_g} is crossed out

with 1 line;
If $d_{k_i,l_j,h_g} = 2$, then s_{k_i,l_j,h_g} element is covered with 2 lines.
We create the IMs $CC[r_0, L*, h_g]$ and $RC[k_i, e_0, h_g]$, in which is recorded that some indexes k_i $(i = 1, \ldots, m)$ of K at time h_g, or l_j $(j = 1, \ldots, n)$ of L at fixed h_g are covered with a line in the matrix S.
for $g = 1$ to f
for $i = 1$ to m
for $j = 1$ to n
– If $\{s_{k_i,l_j,h_g} = \langle 0, 1\rangle$ (or $\langle k_i, l_j, h_g\rangle \in Index_{(\min \mu),k_i,h_g}(S)$, $rm_{k_i,pu,h_g} = 0$ and $d_{k_i,l_j,h_g} = 0\}$, then $\{rc_{k_i,e_0,h_g} = 1; d_{k_i,l_j,h_g} = 1\ \forall l_j;\ pr_{K*,L*,h_g}S_{(k_i,\perp,\perp)}\}$
– If $\{s_{k_i,l_j,h_g} = \langle 0, 1\rangle$ (or $\langle k_i, l_j, h_g\rangle \in Index_{(\min \mu),l_j,h_g}(S)$, $cm_{pu_1,l_j,h_g} = 0$ and $d_{k_i,l_j,h_g} = 1\}$, then $\{cc_{r_0,l_j,h_g} = 1; d_{k_i,l_j,h_g} = 2\ \forall k_i;\ pr_{K*,L*,h_g}S_{(\perp,l_j,\perp)}\}$.
Step 8. Develop the new revised cost IM We select the minimum IF cost of the S at time h_g using the relations (1), that is not crossed by the lines in *Step 7*, and subtract it from each of its uncovered elements at time h_g, and we add it to each of its elements that is covered by two lines at time h_g. We return to *Step 6*.
for $g = 1$ to f
$\{\ AGIndex_{(\min,\max)}\left(pr_{K*,L*,h_g}S\right) = \langle k_x, l_y, h_g\rangle;$
(that finds the smallest element index among the elements of the $pr_{K*,L*,h_g}S$ matrix.)
Subtract S_{k_x,l_y,h_g} uncrossed each element of the IM $pr_{K*,L*,h_g}S$ with reduced prices:

$$IO_{-(\max,\min)}\left(\langle S\rangle, \langle k_x, l_y, h_g pr_{K*,L*,h_g}S\rangle\right).$$

We add it to each element of $pr_{K*,L*,h_g}S$, which is crossed out by two lines, i.e. $d_{k_i,l_j,h_g} = 2$:
for $i = 1$ to m
for $j = 1$ to n
$\{$if $d_{k_i,l_j,h_g} = 2$ then create

$$S_1 = pr_{k_x,l_y,h_g}C;\ S_2 = pr_{k_i,l_j,h_g}C \oplus_{(\max,\min)} \left[\frac{k_i}{k_x}; \frac{l_j}{l_y}; \perp\right] S_1;\ S := S \oplus_{(\max,\min)} S_2;$$

if $d_{k_i,l_j,h_g} = 1$ then $S := S \oplus_{(+)} pr_{k_i,l_j,h_g}C\}$. $\}$ Go to *Step 5*.
Step 9. Determination of a cell for allocation
For $g = 1$ to f
$\{$ **(1)** Use relations (1) to select the largest IF cost in the IM $pr_{K*,L*,h_g}S$ at time $h_g (g = 1, \ldots, f)$. If a tie exists, use any arbitrary tie-breaking choice. Let us denote this cell as c_{k_{x*},l_{y*},h_g}.

$$AGIndex_{(\max,\min)}\left(pr_{K*,L*,h_g}\right) = \langle k_{x*}, l_{y*}, h_g\rangle;$$

(2) Select a single cost with zero degree of membership for allocation corresponding to k_{x*}-th producer and/or l_{y*}-th consumer at time h_g if exists and assigns the most possible to that cost cell and strike the satisfied IF supply or IF demand.

Let us $s_{k_e,l_e*,h_g} = min(s_{Index_{(\min\mu),k_{x*}(pr_{K*,L*,h_g})}}, s_{Index_{(\min\mu),l_{y*}}})(pr_{K*,L*,h_g})$.
Then the minimum of the required and offered quantity is assigned to the corresponding s_{k_e,l_e*,h_g} cell and delete the row/column with exhausted required or offered quantity at time h_g for $(g = 1, \ldots, f)$. So we find the reduced IM S.
We find the minimum between the elements of s_{k_e,R,h_g} and s_{Q,l_e*,h_g} by the operations as follows:
We create IMs $S_8[k_e, R, h_g] = pr_{k_e,R,h_g}S$ and $S_9[Q, l_e*, h_g] = pr_{Q,l_e*,h_g}S$;
If $S_8 \subseteq_v \left[\frac{k_e}{l_g}; \frac{R}{Q}; \perp\right](S_9)'$ (i.e. $\min(s_{k_e,R,h_g}, s_{Q,l_e*,h_g}) = s_{k_e,R,h_g}$), then

$$\{X := X \oplus_{(\max,\min)} \left[\perp; \frac{l_g}{R}; \perp\right] S_8;$$

We obtain a new form of IM $pr_{K*,L*,h_g}S$ with dimensions $(m+2) \times (n+2) \times 1$ by deleting the k_e-th index of $K*$ for $pr_{K*,L*,h_g}S$ using the operation "reduction" $pr_{K*,L*,h_g}S_{(k_e,\perp,\perp)}$.
Let us create IM S_{10} as follows:
$S_{10}[Q, l_e*, h_g] = S_9 -_{(\max,\min)} \left[\frac{Q}{R}; \frac{l_e*}{k_e}; \perp\right](S_8)'$;
Then $S := S \oplus_{(\max,\min)} S_{10}$; }
If $S_8 \supseteq_v \left[\frac{k_i}{l_j}; \frac{R}{Q}; \perp\right](S_9)'$ (i.e. $\min(s_{k_i,R,h_g}, s_{Q,l_j,h_g}) = s_{Q,l_j,h_g}$), then {the IM X changes with $X := X \oplus_{(\max,\min)} \left[\frac{k_e}{Q}; \perp; \perp\right] S_9$.
We obtain a new form of $pr_{K*,L*,h_g}S$ with dimensions $(m+3) \times (n+1) \times 1$ by reduction of the l_{e*}-th column of $pr_{K*,L*,h_g}S$ at time h_g. Let us construct IM S_{11} as follows:
$S_{11}[k_e, R, h_g] = S_8 -_{(\max,\min)} \left[\frac{k_e}{l_{e*}}; \frac{R}{Q}; \perp\right](S_9)'$; $S := S \oplus_{(\max,\min)} S_{11}$; Repeat *Steps (1) and (2)* until $|pr_{K,L,h_g}S| = 6$ (all the required quantities are satisfied and all the offered quantities are exhausted at time h_g), i.e. $pr_{K*,L*,h_g}$ is reduced to the form $S[K^r, L^r, h_g]$

$$= \begin{array}{c|cc} h_g & R & pu \\ \hline Q & \langle \mu_{Q,R,h_g}, \nu_{Q,R,h_g} \rangle & \langle \mu_{Q,pu,h_g}, \nu_{Q,pu,h_g} \rangle \\ & & \\ pl & \langle \mu_{pl,R,h_g}, \nu_{pl,R,h_g} \rangle & \langle \mu_{pl,pu,h_g}, \nu_{pl,pu,h_g} \rangle \\ pu_1 & \langle \mu_{pu_1,R,h_g}, \nu_{pu_1,R,h_g} \rangle & \langle \mu_{pu_1,pu,h_g}, \nu_{pu_1,pu,h_g} \rangle \end{array};$$

} Go to *Step 10*.
Step 10. For $g = 1$ to f
{ $DI = Index_{\not\perp} pr_{K*,L*,h_g}X = \{\langle k_{i*_1}, l_{j*_1}, h_g\rangle, \ldots, \langle k_{i*_f}, l_{j*_f}, h_g\rangle, \ldots, \langle k_{i*_\varphi}, l_{j*_\varphi}, h_g\rangle\}$. If the intuitionistic fuzzy feasible solution at time h_g is degenerated (it contains less than $(m+n-1)$ (the total number of producers and consumers decreased by 1 at time h_g) occupied cells in the $pr_{K*,L*,h_g} = X$ i.e. $|DI| < (m+n-1)$) [4] then increase the basic cells x_{k_i,l_j,h_g} of $pr_{K*,L*,h_g}X$ with one to which the minimum transportation cost corresponds at time h_g. Let us the recorded delivery of this cell is $\langle 0, 1\rangle$. The IMs operations are:

If $|DI| < (m+n-1)$ then
$\{AGIndex_{\{(\min/\max)/(\min_\square/\max_\square)/(\min_\diamond/\max_\diamond)(\min_R/\max_R)\}(\not\perp)(\notin DI)}\left(pr_{K*,L*,h_g}C\right)$
$= \langle k_\alpha, l_\beta, h_g\rangle; x_{k_{al},l_\beta,h_g} = \langle 0, 1\rangle\}.\ \}$
Go to *Step 11*.
Step 11.
for $g = 1$ to f
for $i = 1$ to m
for $j = 1$ to n
{If ($\langle k_i, l_j, h_g\rangle \in EG$ and $x_{k_i,l_j,h_g} \neq \langle\perp,\perp,\perp\rangle$) then {the problem has not solution [4] and the algorithm stop, go to *Step 11*}

Then all the required and offered quantities are exhausted and the algorithm stop. The optimal basic solution $X_{opt}[K*, L*, H*, \{x_{k_i,l_j,h_g}\}]$ is obtained.
for $g = 1$ to f
for $i = 1$ to m
for $j = 1$ to n
If $x_{k_i,l_j,h_g} = \langle\perp,\perp\rangle$ then $x_{k_i,l_j,h_g} = \langle 0.00, 1.00\rangle$.
The optimal intuitionistic fuzzy transportation cost is:

$$AGIO^1_{\oplus_{(\max,\min)}}\left(C_{(\{Q,pl,pu_1\},\{R,pu\},T)} \otimes_{(\min,\max)} X_{opt}\right)$$

$$\text{or } AGIO^2_{\oplus_{(\vee_2)}}\left(C_{(\{Q,pl,pu_1\},\{R,pu\},T)} \otimes_{(\wedge_2)} X_{opt}\right),$$

where $\vee_2$ and $\wedge_2$ are the operations from (2).
Step 12. End.

4 An Application of 3-D Intuitionistic Fuzzy Zero-Point Approach

A trader supplies a product to 4 different companies $\{l_1, l_2, l_3, l_4\}$ in a time-moment $h_g (g = 1, 2, 3)$ after delivery of that product from different m manifacturers (producers) $\{k_1, k_2, k_3\}$ in quantities c_{k_i,R,h_g} (for $i = 1, 2, 3$; $g = 1, 2, 3$). Let the companies $\{l_1, l_2, l_3, l_4\}$ demand this product in a quantity of c_{Q,l_j,h_g} (for $j = 1, 2, 3, 4$; $g = 1, 2, 3$) at time h_g and c_{pl,l_j,h_g} (for $1 \le j \le 4$) are intuitionistic fuzzy limits to the transportation costs of delivery a particular product from the k_i-th source to the l_j-th destination. The trader is not certain about the transportation costs at time h_g, the demanded and supplied quantities due to several uncertainties. Let the cost c_{k_i,l_j,h_g} for transporting one unit quantity of the product from the k_i-th producer to the l_j-th user at time h_g is an IFP and is an element of IFIM $C[K, L, H]$. x_{k_i,l_j,h_g}—the number of units of the product, transported from the k_i-th source to the l_j-th destination at time h_g (for $1 \le i \le 3$, $1 \le j \le 4$ and $1 \le g \le 3$) and is an element of IFIM X with initial elements $\langle\perp,\perp\rangle$. IFPs c_{pl,l_j,h_g} (for $j = 1, \ldots, n$; $g = 1, \ldots, f$)) are limits for the transportation costs of a delivery a product from the k_i-th manifacturer to the l_j-th destination for h_g. $c_{k_i,l_j,T}$—the quantity of units of the product transported by the k_i-th producer to the l_j-th consumer for the defined time period. The priorities of

consumer requests will be defined in advance in IM $PREF$. The trader wants to satisfy the required quantities of the users so that the intuitionistic fuzzy transportation cost is minimum. The intitial form of the cost IM $C[K, L, H/\{T\}]$ is:

h_1	l_1	l_2	l_3	l_4	R	pu
k_1	$\langle 0.6, 0.2\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.3, 0.1\rangle$	$\langle 0.8, 0.1\rangle$	$\langle 0.5, 0.2\rangle$	$\langle \perp, \perp\rangle$
k_2	$\langle 0.5, 0.3\rangle$	$\langle 0.4, 0.1\rangle$	$\langle 0.5, 0.1\rangle$	$\langle 0.3, 0.2\rangle$	$\langle 0.7, 0.1\rangle$	$\langle \perp, \perp\rangle$
k_3	$\langle 0.4, 0.2\rangle$	$\langle 0.3, 0.2\rangle$	$\langle 0.6, 0.1\rangle$	$\langle 0.7, 0.2\rangle$	$\langle 0.4, 0.5\rangle$	$\langle \perp, \perp\rangle$
Q	$\langle 0.4, 0.2\rangle$	$\langle 0.5, 0.3\rangle$	$\langle 0.6, 0.2\rangle$	$\langle 0.06, 0.02\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pl	$\langle 0.65, 0.3\rangle$	$\langle 0.6, 0.4\rangle$	$\langle 0.75, 0.2\rangle$	$\langle 0.75, 0.1\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pu_1	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$

h_2	l_1	l_2	l_3	l_4	R	pu
k_1	$\langle 0.8, 0.1\rangle$	$\langle 0.28, 0.55\rangle$	$\langle 0.24, 0.70\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.7, 0.1\rangle$	$\langle \perp, \perp\rangle$
k_2	$\langle 0.5, 0.4\rangle$	$\langle 0.32, 0.55\rangle$	$\langle 0.32, 0.60\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.5, 0.2\rangle$	$\langle \perp, \perp\rangle$
k_3	$\langle 0.7, 0.3\rangle$	$\langle 0.28, 0.60\rangle$	$\langle 0.20, 0.65\rangle$	$\langle 0.6, 0.2\rangle$	$0.4, 0.5$	$\langle \perp, \perp\rangle$
Q	$\langle 0.5, 0.3\rangle$	$\langle 0.4, 0.2\rangle$	$\langle 0.06, 0.02\rangle$	$\langle 0.6, 0.2\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pl	$\langle 0.82, 0.1\rangle$	$\langle 0.8, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pu_1	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$

h_3	l_1	l_2	l_3	l_4	R	pu
k_1	$\langle 0.7, 0.2\rangle$	$\langle 0.5, 0.2\rangle$	$\langle 0, 12, 0.73\rangle$	$\langle 0.8, 0.2\rangle$	$\langle 0.4, 0.5\rangle$	$\langle \perp, \perp\rangle$
k_2	$\langle 0.6, 0.1\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.10, 0.82\rangle$	$\langle 0.7, 0.2\rangle$	$0.5, 0.2\rangle$	$\langle \perp, \perp\rangle$
k_3	$\langle 0.7, 0.2\rangle$	$\langle 0.5, 0.3\rangle$	$\langle 0.12, 0.76\rangle$	$\langle 0.7, 0.3\rangle$	$\langle 0.7, 0.1$	$\langle \perp, \perp\rangle$
Q	$\langle 0.06, 0.02\rangle$	$\langle 0.4, 0.2\rangle$	$\langle 0.6, 0.2\rangle$	$\langle 0.5, 0.3\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pl	$\langle 0.82, 0.1\rangle$	$\langle 0.8, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pu_1	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$

where $K=\{k_1, k_2, k_3, Q\}$, $L=\{l_1, l_2, l_3, l_4, R, pu\}$, $H/\{T\}=\{h_1, h_2, h_3\}$ and for $1 \le i \le 3$, $1 \le j \le 4$, $1 \le g \le 3$: $\{c_{k_i,l_j,h_g}, c_{k_i,R,h_g}, c_{Q,l_j,h_g}\}$ are IFPs.

We also define

$$X[K*, L*, H*] = \left\{ \begin{array}{c|cccc} h_g \in H & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_2 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_3 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \end{array} \right\},$$

$K*=\{k_1, k_2, k_3\}$, $L*=\{l_1, l_2, l_3, l_4\}$ and $H*=\{h_1, h_2, h_3\}$.

The IM of preferences is

$$PREF[K*, L*, H*] = \left\{ \begin{array}{c|cccc} h_1 & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_2 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_3 & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \end{array} \right.,$$

$$\left.\begin{array}{c|cccc} h_2 & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0.5, 0.4\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_2 & \langle 0.5, 0.4\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_3 & \langle 0.5, 0.4\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \end{array}, \begin{array}{c|cccc} h_3 & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0.5, 0.3\rangle & \langle 0.6, 0, 3\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_2 & \langle 0.5, 0.3\rangle & \langle 0.6, 0, 3\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \\ k_3 & \langle 0.5, 0.3\rangle & \langle 0.6, 0, 3\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \end{array}\right\}.$$

Solution of the problem: **Step 1**. The problem is balanced.
Step 2. Checking the conditions for limiting the transportation costs and the priorities of the requests.

The IM $C[K, L, H/\{T\}]$ is transformed in:

$$\left\{\begin{array}{c|cccccc} h_1 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.6, 0.2\rangle & \langle 1.0, 0.0\rangle & \langle 0.3, 0.1\rangle & \langle 1.0, 0.0\rangle & \langle 0.5, 0.2\rangle & \langle \perp, \perp\rangle \\ k_2 & \langle 0.5, 0.3\rangle & \langle 0.4, 0.1\rangle & \langle 0.5, 0.1\rangle & \langle 0.3, 0.2\rangle & \langle 0.7, 0.1\rangle & \langle \perp, \perp\rangle \\ k_3 & \langle 0.4, 0.2\rangle & \langle 0.3, 0.2\rangle & \langle 0.6, 0.1\rangle & \langle 0.7, 0.2\rangle & \langle 0.4, 0.5\rangle & \langle \perp, \perp\rangle \\ Q & \langle 0.4, 0.2\rangle & \langle 0.5, 0.3\rangle & \langle 0.6, 0.2\rangle & \langle 0.06, 0.02\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.65, 0.3\rangle & \langle 0.6, 0.4\rangle & \langle 0.75, 0.2\rangle & \langle 0.75, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array}\right.,$$

$$\begin{array}{c|cccccc} h_2 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.32, 0.55\rangle & \langle 0.28, 0.55\rangle & \langle 0.24, 0.7\rangle & \langle 0.7, 0.1\rangle & \langle 0.7, 0.1\rangle & \langle \perp, \perp\rangle \\ k_2 & \langle 0.2, 0.7\rangle & \langle 0.32, 0.55\rangle & \langle 0.32, 0.6\rangle & \langle 0.7, 0.1\rangle & \langle 0.5, 0.2\rangle & \langle \perp, \perp\rangle \\ k_3 & \langle 0.28, 0.65\rangle & \langle 0.28, 0.6\rangle & \langle 0.2, 0.65\rangle & \langle 0.6, 0.2\rangle & \langle 0.4, 0.5\rangle & \langle \perp, \perp\rangle \\ Q & \langle 0.5, 0.3\rangle & \langle 0.4, 0.2\rangle & \langle 0.06, 0.02\rangle & \langle 0.6, 0.2\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.82, 0.1\rangle & \langle 0.8, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array},$$

$$\left.\begin{array}{c|cccccc} h_3 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.21, 0.6\rangle & \langle 0.15, 0.68\rangle & \langle 0.12, 0.73\rangle & \langle 0.8, 0.2\rangle & \langle 0.4, 0.5\rangle & \langle \perp, \perp\rangle \\ k_2 & \langle 0.18, 0.55\rangle & \langle 0.21, 0.64\rangle & \langle 0.1, 0.82\rangle & \langle 0.7, 0.2\rangle & \langle 0.5, 0.2\rangle & \langle \perp, \perp\rangle \\ k_3 & \langle 0.21, 0.6\rangle & \langle 0.15, 0.72\rangle & \langle 0.12, 0.76\rangle & \langle 0.7, 0.3\rangle & \langle 0.7, 0.1\rangle & \langle \perp, \perp\rangle \\ Q & \langle 0.06, 0.02\rangle & \langle 0.4, 0.2\rangle & \langle 0.6, 0.2\rangle & \langle 0.5, 0.3\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.82, 0.1\rangle & \langle 0.8, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array}\right\}$$

Let us define IM $S = [K, L, H, \{s_{k_i,l_j,h_g}\}]$ such that $S = C$.
Step 3. Determination of zero membership value—$K*$ level For each index k_i of K at a fixed h_g $(g = 1, \ldots, f)$ of the matrix S, the smallest element is found in accordance with the relations (1): $\langle a, b\rangle \leq_R \langle c, d\rangle$ iff $R_{\langle a,b\rangle} \geq R_{\langle c,d\rangle}$ among the elements s_{k_i,l_j,h_g} $(j = 1, \ldots, n)$ and is saved to the right of the row, in the column pu. It is subtracted from all elements s_{k_i,l_j,h_g}, for $j = 1, 2, \ldots, n$ and go to *Step 4.*
Step 4. Determination of zero membership value—$L*$ level For each index l_j of L at a fixed h_g $(g = 1, 2, 3)$ of the matrix S, the smallest element is found among the elements s_{k_i,l_j,h_g} $(i = 1, 2, 3)$ in accordance with the relation from (1). It is saved at the bottom of the column, in line pu_1 and it is subtracted from all elements s_{k_i,l_j,h_g},

for $i = 1, 2, 3$ and go to *Step 5.*

Step 5. Determination of zero membership value—$H*$ level For each index k_i of K at a fixed l_j $(j = 1, \ldots, n)$ of the matrix S, the smallest element is found among the elements s_{k_i,l_j,h_g} $(g = 1, \ldots, f)$ and is recorded as the value of the element $s_{k_i,l_j,T}$. It is subtracted from all elements s_{k_i,l_j,h_g}, for $g = 1, 2, \ldots, f$ and go to *Step 6.*

The form of the matrix $ST = S[K, L, H/\{T\}]$ takes the following form: after computations of *Steps 4, 5 and 6* is:

$$ST = \left\{ \begin{array}{c|cccccc} h_1 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.0, 0.6\rangle & \langle 0.67, 0.17\rangle & \langle 0.0, 1.0\rangle & \langle 0.64, 0.36\rangle & \langle 0.5, 0.2\rangle & \langle 0.3, 0.1\rangle \\ k_2 & \langle 0.2, 0.8\rangle & \langle 0.0, 0.6\rangle & \langle 0.2, 0.8\rangle & \langle 0.0, 0.8\rangle & \langle 0.7, 0.1\rangle & \langle 0.3, 0.2\rangle \\ k_3 & \langle 0.92, 0.02\rangle & \langle 0.0, 0.8\rangle & \langle 0.3, 0.7\rangle & \langle 0.4, 0.6\rangle & \langle 0.4, 0.5\rangle & \langle 0.3, 0.2\rangle \\ Q & \langle 0.4, 0.2\rangle & \langle 0.5, 0.3\rangle & \langle 0.6, 0.2\rangle & \langle 0.06, 0.02\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.65, 0.3\rangle & \langle 0.6, 0.4\rangle & \langle 0.75, 0.2\rangle & \langle 0.75, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle 0.1, 0.4\rangle & \langle 0.0, 0.4\rangle & \langle 0.0, 0.2\rangle & \langle 0.0, 0.4\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array} \right.,$$

$$\begin{array}{c|cccccc} h_2 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.0, 1.0\rangle & \langle 0.01, 0.99\rangle & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.7, 0.1\rangle & \langle 0.24, 0.7\rangle \\ k_2 & \langle 0.0, 1.0\rangle & \langle 0.02, 0.98\rangle & \langle 0.12, 0.88\rangle & \langle 0.1, 0.9\rangle & \langle 0.5, 0.2\rangle & \langle 0.2, 0.7\rangle \\ k_3 & \langle 0.0, 1.0\rangle & \langle 0.08, 0.92\rangle & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.4, 0.5\rangle & \langle 0.2, 0.65\rangle \\ Q & \langle 0.5, 0.3\rangle & \langle 0.4, 0.2\rangle & \langle 0.06, 0.02\rangle & \langle 0.6, 0.2\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.82, 0.1\rangle & \langle 0.8, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle 0.0, 1.0\rangle & \langle 0.04, 0.96\rangle & \langle 0.0, 1.0\rangle & \langle 0.4, 0.6\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array},$$

$$\left. \begin{array}{c|cccccc} h_3 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.0, 1.0\rangle & \langle 0.0, 0.14\rangle & \langle 0.12, 0.88\rangle & \langle 0.04, 0.96\rangle & \langle 0.4, 0.5\rangle & \langle 0.12, 0.73\rangle \\ k_2 & \langle 0.08, 0.92\rangle & \langle 0.01, 0.99\rangle & \langle 0.0, 1.0\rangle & \langle 0.02, 0.98\rangle & \langle 0.5, 0.2\rangle & \langle 0.1, 0.82\rangle \\ k_3 & \langle 0.01, 0.99\rangle & \langle 0.03, 0.97\rangle & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.7, 0.1\rangle & \langle 0.12, 0.76\rangle \\ Q & \langle 0.06, 0.02\rangle & \langle 0.4, 0.2\rangle & \langle 0.6, 0.2\rangle & \langle 0.5, 0.3\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.82, 0.1\rangle & \langle 0.8, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle 0.85, 0.1\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle 0.08, 0.92\rangle & \langle 0.03, 0.97\rangle & \langle 0.0, 1.0\rangle & \langle 0.58, 0.42\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array} \right\}$$

Step 6. Optimality criterion

(1) Check if each quantity of product offered at time h_g is less than or equal the total quantity of product required, whose reduced costs are with zero membership degrees.

(2) Check if each required quantity at time h_g is less than or equal to the total quantity offered, whose reduced costs have zero membership degrees.

(3) If 6.1 and 6.2 are satisfied then go to *Step 9.* else go to *Step 7.*

Step 7. Revise the cost IM All elements $\langle 0, 1\rangle$ in the S are crossed out with minimum number of lines (horizontal, vertical or both). If there is no element $\langle 0, 1\rangle$ in a given row or column at time h_g (for $g = 1, \ldots, f$), then the element with the minimum degree of membership is crossed out from that row or column in the cost IM S

obtained in *s*tep 6.

Step 8. Develop the new revised cost IM We select the minimum IF cost of the ST at time h_g using the relations (1), that is not crossed by the lines in *Step 7*, and subtract it from each of its uncovered elements at time h_g, and we add it to each of its elements that is covered by two lines at time h_g. We return to *Step 6*.

The *Steps 6., 7. and 8.* are executed twice and then proceeds to *Step 9.* The IM ST takes the following form after these steps:

$$ST = \left\{ \begin{array}{c|cccccc} h_1 & l_1 & l_2 & l_3 & l_4 & R & pu \\ \hline k_1 & \langle 0.0, 1.0\rangle & \langle 0.5, 0.5\rangle & \langle 0.0, 0.4\rangle & \langle 0.6, 0.4\rangle & \langle 0.5, 0.2\rangle & \langle 0.3, 0.1\rangle \\ k_2 & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.28, 0.45\rangle & \langle 0.0, 0.8\rangle & \langle 0.7, 0.1\rangle & \langle 0.3, 0.2\rangle \\ k_3 & \langle 0.0, 0.8\rangle & \langle 0.0, 0.8\rangle & \langle 0.43, 0.41\rangle & \langle 0.46, 0.54\rangle & \langle 0.4, 0.5\rangle & \langle 0.3, 0.2\rangle \\ Q & \langle 0.4, 0.2\rangle & \langle 0.5, 0.3\rangle & \langle 0.6, 0.2\rangle & \langle 0.06, 0.02\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pl & \langle 0.55, 0.3\rangle & \langle 0.6, 0.4\rangle & \langle 0.75, 0.2\rangle & \langle 0.65, 0.3\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \\ pu_1 & \langle 0.1, 0.4\rangle & \langle 0, 0.4\rangle & \langle 0, 0.2\rangle & \langle 0, 0.4\rangle & \langle \perp, \perp\rangle & \langle \perp, \perp\rangle \end{array} \right.$$

h_2	l_1	l_2	l_3	l_4	R	pu
k_1	$\langle 0.02, 0.98\rangle$	$\langle 0.01, 0.99\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.24, 0.7\rangle$
k_2	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.1, 0.9\rangle$	$\langle 0.08, 0.92\rangle$	$\langle 0.5, 0.2\rangle$	$\langle 0.2, 0.7\rangle$
k_3	$\langle 0.02, 0.98\rangle$	$\langle 0.08, 0.92\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.4, 0.5\rangle$	$\langle 0.2, 0.65\rangle$,
Q	$\langle 0.5, 0.3\rangle$	$\langle 0.4, 0.2\rangle$	$\langle 0.06, 0.02\rangle$	$\langle 0.6, 0.2\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pl	$\langle 0.82, 0.1\rangle$	$\langle 0.8, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pu_1	$\langle 0.0, 1.0\rangle$	$\langle 0.04, 0.96\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.4, 0.6\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$

h_3	l_1	l_2	l_3	l_4	R	pu
k_1	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 0.14\rangle$	$\langle 0.13, 0.8\rangle$	$\langle 0.04, 0.96\rangle$	$\langle 0.4, 0.5\rangle$	$\langle 0.12, 0.73\rangle$
k_2	$\langle 0.07, 0.93\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.01, 0.99\rangle$	$\langle 0.5, 0.2\rangle$	$\langle 0.1, 0.82\rangle$
k_3	$\langle 0.01, 0.99\rangle$	$\langle 0.03, 0.97\rangle$	$\langle 0.01, 0.99\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.7, 0.1\rangle$	$\langle 0.12, 0.76\rangle$
Q	$\langle 0.06, 0.02\rangle$	$\langle 0.4, 0.2\rangle$	$\langle 0.6, 0.2\rangle$	$\langle 0.5, 0.3\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pl	$\langle 0.82, 0.1\rangle$	$\langle 0.8, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle 0.85, 0.1\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$
pu_1	$\langle 0.08, 0.92\rangle$	$\langle 0.03, 0.97\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.58, 0.42\rangle$	$\langle \perp, \perp\rangle$	$\langle \perp, \perp\rangle$

Step 9. Determination of a cell for allocation

For $g = 1$ to f

{ **(1)** Use relations (1) to select the largest IF cost in the IM $pr_{K*,L*,h_g}S$ at time $h_g (g = 1, 2, 3)$. Let us denote this cell as c_{k_{x*},l_{y*},h_g}. **(2)** Select a single cost with zero degree of membership for allocation corresponding to k_{x*}-th index and/or l_{y*}-th at time h_g if exists and assigns the most possible to that cost cell and strike the satisfied IF supply or IF demand.

(2) Then the minimum of the required and offered quantity is assigned to the corresponding s_{k_e,l_e*,h_g} cell and delete the row/column with exhausted required or offered quantity at time h_g for $(g = 1, 2, 3)$.

Repeat *Steps (1) and (2)* until $|pr_{K,L,h_g}S| = 6$ (all the required quantities are satisfied and all the offered quantities are exhausted at time h_g). }

Step 10. The intuitionistic fuzzy optimal solution, presented by the IM X_{opt} is non-degenerated, it includes 6×3 occupied cells. The optimal intuitionistic fuzzy optimal solution $X_{opt}[K*, L*, H*, \{x_{k_i,l_j,h_g}\}]$ is obtained and has the following form:

$$X_{opt} = \left\{ \begin{array}{c|cccc} h_1 & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0.4, 0.2\rangle & \langle 0.0, 1.0\rangle & \langle 0.1, 0.4\rangle & \langle 0.0, 1.0\rangle \\ k_2 & \langle 0.0, 1.0\rangle & \langle 0.5, 0.3\rangle & \langle 0.14, 0.42\rangle & \langle 0.06, 0.02\rangle \\ k_3 & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.4, 0.5\rangle & \langle 0.0, 1.0\rangle \end{array} \right. \tag{3}$$

h_2	l_1	l_2	l_3	l_4
k_1	$\langle 0.0, 1.0\rangle$	$\langle 0.4, 0.6\rangle$	$\langle 0.06, 0.02\rangle$	$\langle 0.2, 0.7\rangle$
k_2	$\langle 0.5, 0.3\rangle$	$\langle 0.0, 0.5\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$
k_3	$\langle 0.0, 1.0\rangle$	$\langle\langle 0.0, 1.0\rangle$	$\langle 0.0, 1.0\rangle$	$\langle 0.4, 0.5\rangle$

$$\left. \begin{array}{c|cccc} h_3 & l_1 & l_2 & l_3 & l_4 \\ \hline k_1 & \langle 0.06, 0.22\rangle & \langle 0.34, 0.52\rangle & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle \\ k_2 & \langle 0.0, 1.0\rangle & \langle 0.0, 1.0\rangle & \langle 0.5, 0.2\rangle & \langle 0.0, 1.0\rangle \\ k_3 & \langle 0.0, 1.0\rangle & \langle 0.06, 0.72\rangle & \langle 0.1, 0.4\rangle & \langle 0.5, 0.3\rangle \end{array} \right\}$$

Step 11. The optimal intuitionistic fuzzy transportation cost is:

$$AGIO^1_{\oplus(\max,\min))}\left(C_{(\{Q,pl,pu_1\},\{R,pu\},T)} \otimes_{(\min,\max)} X_{opt}\right) = \langle 0.50, 0.42\rangle \tag{4}$$

The degree of membership (acceptance) of this optimal solution is equal to 0.50 and the its degree of non-membership (non-acceptance) is equal to 0.42.

The example illustrates the reliability of the proposed 3-D IFZPM.

5 Conclusion

In this paper it is proposed for the first time to extend the IFZPM [63] to 3-D IFZPM for finding an optimal solution of a type of 3-D IFTP using the concepts of the IMs anf IFSs. The formulated IFTP has additional constraints: upper limits to the transportation costs and supply preferences for consumers. The proposed algorithm for solution of the 3-D IFTP is illustrated with a numerical example. The advantages of the proposed algorithm is that it can be easy generalized to the multidimensional intuitionistic fuzzy TPs [10] and also can be applied to both the TP with crisp parameters and with intuitionistic fuzzy ones. In the future, we will extend 3-D IFZPM to the multidimensional intuitionistic fuzzy TPs [10] and will apply the proposed approach for various types multidimensional IF optimization problems.

References

1. Ahlatcioglu, M., Sivri, M., Guzel, N.: Transportation of the fuzzy amounts using the fuzzy cost. J. Marmara Pure Appl. Sci. **18**, 141–157 (2002)
2. Antony, R., Savarimuthu, S., Pathinathan, T.: Method for solving the transportation problem using triangular intuitionistic fuzzy number. Int. J. Comput. Algorithm **03**, 590–605 (2014)
3. Arsham, H., Khan, A.: A simplex type algorithm for general transportation problems-an alternative to stepping stone. J. Oper. Res. Soc. **40**(6), 581–590 (2017)
4. Atanassov, B.: Quantitative Methods in Business Management. Publ. house TedIna, Varna (1994).(in Bulgarian)
5. Atanassov, K.T.: Intuitionistic fuzzy sets, VII ITKR Session, Sofia, 20–23 June 1983 (Deposed in Centr. Sci.-Techn. Library of the Bulg. Acad. of Sci., 1697/84) (in Bulgarian). Reprinted: Int. J. Bioautomation **20**(S1), S1–S6 (2016)
6. Atanassov, K.: Generalized index matrices. Comptes rendus de l'Academie Bulgare des Sci. **40**(11), 15–18 (1987)
7. Atanassov, K.: On Intuitionistic Fuzzy Sets Theory. STUDFUZZ, vol. 283. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29127-2
8. Atanassov, K.: Index Matrices: Towards an Augmented Matrix Calculus. Studies in Computational Intelligence, vol. 573. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10945-9
9. Atanassov, K.: Intuitionistic Fuzzy Logics. Studies in Fuzziness and Soft Computing, vol. 351. Springer (2017). https://doi.org/10.1007/978-3-319-48953-7
10. Atanassov, K.: n-Dimensional extended index matrices Part 1. Adv. Stud. Contemp. Math. **28**(2), 245–259 (2018)
11. Atanassov, K., Szmidt, E., Kacprzyk, J.: On intuitionistic fuzzy pairs. Notes Intuit. Fuzzy Sets **19**(3), 1–13 (2013)
12. Atanassov, K.: Remark on an intuitionistic fuzzy operation "division". Annual of "Informatics". Section, Union of Scientists in Bulgaria, vol. 10 (2019) (in press)
13. Basirzadeh, H.: An approach for solving fuzzy transportation problem. Appl. Math. Sci. **5**, 1549–1566 (2011)
14. Chanas, S., Kolodziejckzy, W., Machaj, A.: A fuzzy approach to the transportation problem. Fuzzy Sets Syst. **13**, 211–221 (1984)
15. Chanas, S., Kuchta, D.: A concept of the optimal solution of the transportation problem with fuzzy cost coefficients. Fuzzy Sets Syst. **82**, 299–305 (1996)
16. Dantzig, G.: Application of the simplex method to a transportation problem. Chapter XXIII, Activity analysis of production and allocation, New York, Wiley, Cowles Commision Monograph **13**, 359–373 (1951)
17. Das, M., Baruah, H.: Solution of the transportation problem in fuzzified form. J. Fuzzy Math. **15**, 79–95 (2007)
18. De, S.K., Bisvas, R., Roy, R.: Some operations on IFSs. Fuzzy Sets Syst. **114**(4), 477–484 (2000)
19. Dhanasekar, S., Hariharan, S., Sekar, P.: Fuzzy Hungarian MODI Algorithm to solve fully fuzzy transportation problems. Int. J. Fuzzy Syst. **19**(5), 1479–1491 (2017)
20. Dinagar, D., Palanivel, K.: On trapezoidal membership functions in solving transportation problem under fuzzy environment. Int. J. Comput. Phys. Sci. **1**, 1–12 (2009)
21. Gani, A., Samuel, A., Anuradha, D.: Simplex type algorithm for solving fuzzy transportation problem. Tamsui Oxf. J. Inf. Math. Sci. **27**, 89–98 (2011)
22. Gani, A.N., Abbas, S.: Solving intuitionistic fuzzy transportation problem using zero suffix algorithm. Int. J. Math. Sci. Eng. Appl. **6**(3), 73–82 (2012)
23. Gani, A., Abbas, S.: A new average method for solving intuitionistic fuzzy transportation problem. Int. J. Pure Appl. Math. **93**(4), 491–499 (2014)
24. Gen, M., Ida, K., Li, Y., Kubota, E.: Solving bicriteria solid transportation problem with fuzzy numbers by a genetic algorithm. Comput. Ind. Eng. **29**(1), 537–541 (1995)

25. Gupta, G., Kumar, A., Sharma, M.: A note on a new method for solving fuzzy linear programming problems based on the fuzzy linear complementary problem (FLCP). Int. J. Fuzzy Syst. 1–5 (2016)
26. Guzel, N.: Fuzzy transportation problem with the fuzzy amounts and the fuzzy costs. World Appl. Sci. J. **8**, 543–549 (2010)
27. Hitchcock, F.: The distribution of a product from several sources to numerous localities. J. Math. Phys. **20**, 224–230 (1941)
28. Jahirhussain, R., Jayaraman, P.: Fuzzy optimal transportation problem by improved zero suffix method via robust rank techniques. Int. J. Fuzzy Math. Syst. (IJFMS) **3**(4), 303–311 (2013)
29. Jahirhussain, R., Jayaraman, P.: A new method for obtaining an optinal solution for fuzzy transportation problems. International Journal of Mathematical Archive. **4**(11), 256–263 (2013)
30. Jimenez, F., Verdegay, J.: Solving fuzzy solid transportation problems by an evolutionary algorithm based parametric approach. Eur. J. Oper. Res. **117**(3), 485–510 (1999)
31. Sharma, M.K.: Zero-point maximum allocation method for solving intuitionistic fuzzy transportation problem. Int. J. Appl. Comput. Math **6**, 115 (2020). https://doi.org/10.1007/s40819-020-00867-6
32. Kantorovich, L., Gavyrin, M.: Application of mathematical methods in the analysis of cargo flows, Coll. of articles Problems of increasing the efficiency of transport, M.: Publ. house AHSSSR, 110–138 (1949) (in Russian)
33. Karthy, T., Ganesan, K.: Revised improved zero point method for the trapezoidal fuzzy transportation problems. In: AIP Conference Proceedings, vol. 2112, no. 020063, pp. 1–8 (2019)
34. Kathirvel, K., Balamurugan, K.: Method for solving fuzzy transportation problem using trapezoidal fuzzy numbers. Int. J. Eng. Res. Appl. **2**(5), 2154–2158 (2012)
35. Kathirvel, K., Balamurugan, K.: Method for solving unbalanced transportation problems using trapezoidal fuzzy numbers. Int. J. Eng. Res. Appl. **3**(4), 2591–2596 (2013)
36. Kaur, A., Kumar, A.: A new approach for solving fuzzy transportation problems using generalized trapezoidal fuzzy numbers. Appl. Soft Comput. **12**(3), 1201–1213 (2012)
37. Kaur, A., Kacprzyk, J., Kumar, A.: Fuzzy transportation and transshipment problems. Stud. Fuziness Soft Comput. **385** (2020)
38. Kikuchi, S.: Amethod to defuzzify the fuzzy number: transportation problem application. Fuzzy Sets Syst. **116**, 3–9 (2000)
39. Kumar, P., Hussain, R.: A method for solving unbalanced intuitionistic fuzzy transportation problems. Notes Intuit. Fuzzy Sets **21**(3), 54–65 (2015)
40. Lalova, N., Ilieva, L., Borisova, S., Lukov, L., Mirianov, V.: A Guide to Mathematical Programming. Science and Art Publishing House, Sofia (1980).(in Bulgarian)
41. Liang, T., Chiu, C., Cheng, H.: Using possibilistic linear programming for fuzzy transportation planning decisions. Hsiuping J. **11**, 93–112 (2005)
42. Liu, S., Kao, C.: Solving fuzzy transportation problems based on extension principle. Eur. J. Oper. Res. **153**, 661–674 (2004)
43. Liu, S., Kao, C., Network flow problems with fuzzy arc lengths. IEEE Trans. Syst. Man Cybern. Part B Cybern. **34**, 765–769 (2004)
44. Mohideen, S., Kumar, P.: A comparative study on transportation problem in fuzzy environment. Int. J. Math. Res. **2**(1), 151–158 (2010)
45. Oheigeartaigh, M.: A fuzzy transportation algorithm. Fuzzy Sets Syst. **8**, 235–243 (1982)
46. Pandian, P., Natarajan, G.: A new algorithm for finding a fuzzy optimal solution for fuzzy transportation problems. Appl. Math. Sci. **4**, 79–90 (2010)
47. Pandian, P., Natarajan, G.: An optimal more-for-less solution to fuzzy transportation problems with mixed constraints. Appl. Math. Sci. **4**, 1405–1415 (2010)
48. Patil, A., Chandgude, S.: Fuzzy Hungarian approach for transportation model. Int. J. Mech. Ind. Eng. **2**(1), 77–80 (2012)
49. Purushothkumar, M., Ananthanarayanan, M., Dhanasekar, S.: Fuzzy zero suffix algorithm to solve fully fuzzy transportation problems. Int. J. Pure Appl. Math. **119**(9), 79–88 (2018)
50. Riecan, B., Atanassov, A.: Operation division by n over intuitionistic fuzzy sets. NIFS **16**(4), 1–4 (2010)

51. Saad, O., Abbas, S.: A parametric study on transportation problem under fuzzy environment. J. Fuzzy Math. **11**, 115–124 (2003)
52. Samuel, A.: Improved zero point method. Appl. Math. Sci. **6**(109), 5421–5426 (2012)
53. Samuel, A., Venkatachalapathy, M.: Improved zero point method for unbalanced FTPs. Int. J. Pure Appl. Math. **94**(3), 419–424 (2014)
54. Samuel, A., Venkatachalapathy, M.: Modified vogel's approximation method for fuzzy transportation problems. Appl. Math. Sci. **5**, 1367–1372 (2011)
55. Shanmugasundari, M., Ganesan, K.: A novel approach for the fuzzy optimal solution of fuzzy transportation problem. Int. J. Eng. Res. Appl. **3**(1), 1416–1424 (2013)
56. Smimou, K.: A fuzzy transshipment model for allocating foreign currencies. Int. J. Oper. Res. **2**, 284–307 (2007)
57. Sudhagar, V., Navaneethakumar, V.: Solving the multi objective two stage fuzzy transportation problem by zero suffix method. J. Math. Res. **2**(4), 135–140 (2010)
58. Szmidt, E., Kacprzyk, J.: Amount of information and its reliability in the ranking of Atanassov's intuitionistic fuzzy alternatives. In: Rakus-Andersson, E., Yager, R., Ichalkaranje, N., Jain, L.C. (eds.) Recent Advances in Decision Making, SCI. Springer, Heidelberg, vol. 222, pp. 7–19 (2009). https://doi.org/10.1007/978-3-642-02187-9_2
59. Traneva, V.: On 3-dimensional intuitionistic fuzzy index matrices. Notes Intuit. Fuzzy Sets **20**(4), 59–64 (2014)
60. Traneva, V.: Internal operations over 3-dimensional extended index matrices. In: Proceedings of the Jangjeon Mathematical Society, vol. 18, no. 4, pp. 547–569 (2015)
61. Traneva, V., Tranev, S.: Index Matrices as a Tool for Managerial Decision Making. Publ. House of the Union of Scientists, Bulgaria (2017).(in Bulgarian)
62. Traneva, V., Tranev, S.: Index Matrices in the Assessment of Human Resources. Publ. House of the Union of Scientists, Bulgaria (2019)
63. Traneva V., Tranev S.: Intuitionistic fuzzy transportation problem by zero point method. In: Proceedings of the 15th Conference on Computer Science and Information Systems (FedCSIS). Sofia, Bulgaria, pp. 345–348 (2020). https://doi.org/10.15439/2020F61
64. Traneva, V., Tranev, S.: An Intuitionistic fuzzy zero suffix method for solving the transportation problem. In: Dimov I., Fidanova S. (eds.) Advances in High Performance Computing. HPC 2019. Studies in Computational Intelligence, vol. 902. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-55347-0_7
65. Traneva, V., Tranev, S., Atanassova, V.: An intuitionistic fuzzy approach to the hungarian algorithm. In: Nikolov G., et al. (eds.) Springer Nature Switzerland AG, NMA 2018, LNCS 11189, pp. 1–9 (2019). https://doi.org/10.1007/978-3-030-10692-8_19
66. Traneva, V., Tranev, S., Stoenchev, M., Atanassov, K.: Scaled aggregation operations over two- and three-dimensional index matrices. Soft Comput. **22**, 5115–5120 (2019). https://doi.org/10.1007/00500-018-3315-6
67. Zadeh, L.: Fuzzy sets. Inf. Control **8**(3), 338–353 (1965)

On Index-Matrix Interpretation of Interval-Valued Intuitionistic Fuzzy Hamiltonian Cycle

Velichka Traneva and Stoyan Tranev

Abstract A Hamiltonian cycle in a graph is a cycle that visits each node/vertex exactly once. There are different approaches to find the Hamiltonian cycle in a graph. Graph theory finds application in some optimization problems like traveling salesman problem, resource networking and etc. Fuzzy graph theory is finding an increasing number of applications in modeling real data systems where the level of information fundamental in the system varies with different levels of precision. Zadeh's fuzzy theory emerged in 1963 to deal with the uncertainty in the environment and with the information noise. The values of membership and non-membership cannot handle such uncertainty and hesitation involved in real-life problems. Thus Atanassov and Gargov in 1989 first proposed the concept of interval-valued intuitionistic fuzzy sets (IVIFSs). In this paper, we propose for the first time an index-matrix interpretation of Hamiltonian cycle in an Interval-Valued Intuitionistic Fuzzy graph (IVIFG), based on the theories of IVIFSs and of index matrices (IMs). The aim of the paper is to propose an extension of the algoirithms to find an intuitionistic fuzzy Hamiltonian cycle (IFHC) in an IFG to the interval-valued intuitionistic fuzzy Hamiltonian cycle (IVIFHC) using the IVIFSs and IMs concepts. The algorithm for finding Hamiltonian cycle in IVIFG is applied over real IVIFG example about network of Wizz air airlines. In the paper also are extended three index-type operations to the case of interval-valued intuitionistic fuzzy IMs (IVIFIMs).

The work on Sect. 1 and Sect. 2 is supported by the Bulgarian National Scientific program "ICT in science, education and security". The work on Sect. 3 and Sect. 4 is supported by the Bulgarian Ministry of Education and Science under the National Research Program "Young scientists and postdoctoral students", approved by DCM # 577/ 17.08.2018.

V. Traneva (✉) · S. Tranev
"Prof. Asen Zlatarov" University, "Prof. Yakimov" Blvd, 8000 Bourgas, Bulgaria
e-mail: tranev@abv.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_17

1 Introduction

The first problem of graph theory known as koningsberg bridge problem is considered in [13]. A graph theory has many applications in different areas of computer science. A Hamiltonian cycle, which is named after Sir William Rowan Hamilton, is a cycle through a graph that visits each node exactly once (see [24]). Determining if a graph is Hamiltonian is well known to be NP-complete [17]. Dirac (1952, [11]) described some relations between the degree of the nodes in a graph and the lengths of the circuits contained in it. Ore, Chvatal and Fan have provided the sufficient conditions for a graph to be Hamiltonian (see [10, 14, 20]). Rahman and Kaykobad, Zhao, in 2007, gave better conditions for the existence of Hamiltonian paths in a graph (see [35]). The Hamiltonicity of graphs was researched by Lee and Sudakov [18], Shang [22], Dudek and Ferrara [12], and etc.

Nowadays, some parameters of a graph problem may be uncertain due to uncontrollable factors. Zadeh, in [34], is proposed the idea of a fuzzy set as a class of objects with unclear boundaries, i.e. in which elements may belong to partial degrees, from the full belongingness to the full non-belongingness through all intermediate values. In 1983, Atanassov proposed the IFSs [1], which is an extension of FSs of Zadeh. IFSs are a more flexible tool for reflecting ambiguity in parameters than fuzzy ones because they have a hesitancy degree. Then Atanassov and Gargov in 1989 proposed the concept of interval-valued intuitionistic fuzzy sets (IVIFSs, [8]), which have interval-valued degrees of membership and non-membership.

In the following, a brief review of the basic research done in the area of fuzzy, intuitionistic-fuzzy and interval-valued intuitionistic fuzzy Hamiltonian graph is presented.

The first idea of a fuzzy graph was given by Kaufman [16]. Rosenfeld [21] developed the theory of fuzzy graphs in 1975. Yeh and Bang, Mordeson and Nair have also introduced various concepts in fuzzy graphs (see [19, 33]). A new algorithm to find fuzzy Hamiltoian cycle in a fuzzy network using adjacency matrix has proposed by Gani and Latha [15]. In [25] generalized fuzzy Hamiltonian graphs have introduced.

In the paper [29], the algorithm for finding Hamiltonian cycle in an intuitionistic fuzzy graph (IFG) is proposed, based on the theories of intuitionistic fuzzy sets (IFSs) and of index matrices (IMs).

In this paper, which is an extension of [29], it is proposed for the first time algorithm for finding Hamiltonian cycle in an interval-valued intuitionistic fuzzy graph (IVIFG, [6]), based on the concepts of IVIFSs [8] and of IMs (see [2, 4]). The proposed approach is applied to an IVIFG example about a network of Wiz Air airlines. The main contributions of the paper is that it proposes for the first time the algorithm for finding Hamiltonian cycle in IVIFG using IVIFSs and IMs concepts and demonstrates the effectiveness of the algorithm for example with real data for a network map of Wizz air airlines [32]. In the paper also are extended three index-type operations to the case of interval-valued intuitionistic fuzzy IMs (IVIFIMs).

The structure of this paper is as follows: Sect. 2 describes the related concepts of the IMs, IVIFPs and IVIFGs. In the section also are introduced for the first time three

index-type operations over IVIFIMs. In Sect. 3, an algorithm to find a Hamiltonian cycle in an IVIFG is described, extending the algorithm of [29] and using the concepts of IMs and IVIFSs. The effectiveness of the IVIF approach for finding a Hamiltonian cycle in a graph is demonstrated by an example with real data in Sect. 4. Section 5 offers the conclusion and outlines some aspects for future research.

2 Basic Definitions of IVIFIMs, Interval-Valued Intuitionistic Fuzzy Pairs and IVIFGs

In this section we recall some basic definitions on interval-valued intuitionistic fuzzy pairs (IVIFPs) from (see [8, 9, 26]) and on index matrix apparatus from (see [4, 7]).

2.1 Short Remarks on IVIFPs

The **IVIFP** is an object of the form $\langle M, N\rangle$, where $M, N \subseteq [0, 1]$, $M = \left[\inf M, \sup M\right]$, $N = \left[\inf N, \sup N\right]$ and $\sup M + \sup N \leq 1$, that are used as an evaluation of some object or process.

The IVIFFP x is an **"interval-valued inuitionistic fuzzy false pair" (IVIFFP)** if and only if $\inf M \leq \sup N$, while x is a **"false pair" (IVFFP)** iff $M = [0, 0]$, $N = [1, 1]$.

Let us have two IVIFPs $x = \langle M, N\rangle$ and $y = \langle P, Q\rangle$. In [5, 9] are defined the operations negation, conjunction, disjunction, multiplication with constant, and difference

$$
\begin{aligned}
&\neg x = \langle N, M\rangle; \\
&x \wedge_1 y = && \langle\left[\min(\inf M, \inf P), \min(\sup M, \sup P)\right], \\
& && \left[\max(\inf N, \inf Q), \max(\sup N, \sup Q)\right]\rangle; \\
&x \vee_1 y && = \langle\left[\max(\inf M, \inf P), \max(\sup M, \sup P)\right], \\
& && \left[\min(\inf N, \inf Q), \min(\sup N, \sup Q)\right]\rangle; \\
&x \wedge_2 y = x + y = && \langle\left[\inf M + \inf P - \inf M \inf P, \sup M + \sup P\right. \\
& && \left.- \sup M \sup P\right], \left[\inf N \inf Q, \sup N \sup Q\right]\rangle; \\
&x \vee_2 y = x.y = && \langle\left[\inf M \inf P, \sup M \sup P\right], [\inf N + \inf Q - \inf N \inf Q, \\
& && \sup N + \sup Q - \sup N \sup Q\right]\rangle; \\
&x - y = && \langle[\max(0, \min(\inf M - \sup P, 1 - \sup N + \sup Q)), \\
& && \max(0, \min(\sup M - \inf P, 1 - \sup N + \sup Q))], \\
& && [\min(1, \min(\inf N + \inf Q, 1 - \sup M + \inf P)), \\
& && \min(1, \sup N + \sup Q, 1 - \sup M + \inf P)]\rangle;
\end{aligned}
\tag{1}
$$

$$
x \&_{139} y = x \vee_{139} y = x @ y = \langle\left[\frac{\inf M + \inf P}{2}, \frac{\sup M + \sup N}{2}\right], \left[\frac{\inf N + \inf Q}{2}, \frac{\sup N + \sup Q}{2}\right]\rangle
$$

and the relations

$$
\begin{array}{lll}
x \le y & \text{iff} & \inf M \le \inf P \text{ and } \sup M \le \sup P \\
 & & \text{and } \inf N \ge \inf Q \text{ and } \sup N \ge \sup Q \\
x \le_{\square} y & \text{iff} & \inf M \le \inf P \text{ and } \sup M \le \sup P \\
x \le_{\diamond} y & \text{iff} & \inf N \ge \inf Q \text{ and } \sup N \ge \sup Q \\
x \le_{R} y & \text{iff} & R_{\langle \inf M, \inf N \rangle} \ge R_{\langle \inf P, \inf Q \rangle} \\
 & & \text{and } R_{\langle \sup M, \sup N \rangle} \ge R_{\langle \sup P, \sup Q \rangle} \\
x = y & \text{iff} & \inf M = \inf P \text{ and } \sup M = \sup P \\
 & & \text{and } \inf Q = \inf N \text{ and } \sup Q = \sup N \\
x =_{\square} y & \text{iff} & \inf M = \inf P \text{ and } \sup M = \sup P \\
x =_{\diamond} y & \text{iff} & \inf N = \inf Q \text{ and } \sup N = \sup Q
\end{array}
\quad , \tag{2}
$$

where $R_{\langle a,b \rangle} = 0.5(2 - a - b)(1 - a)$ [26].

2.2 Definition, Operations and Relations over Extended Interval-Valued Intuitionistic Fuzzy Index Matrices (EIVIFIMs)

2.2.1 Definition of EIVIFIM

Let $\mathscr{I}$ be a fixed set. For the sets $K = \{k_1, \ldots, k_i, \ldots, k_m\}$, $L = \{l_1, \ldots, l_j, \ldots, l_n\}$ $(K, L \subset \mathscr{I})$, a 2-D extended interval-valued intuitionistic fuzzy IM (2-D EIVIFIM) is defined by [7]:

$$[K^*, L^*, \{\langle M_{k_i,l_j}, N_{k_i,l_j} \rangle\}]$$

$$
\equiv \begin{array}{c|cccccc}
 & l_1, \langle A_{l_1}, B_{l_1} \rangle & \ldots & l_j, \langle A_{l_j}, B_{l_j} \rangle & \ldots & l_n, \langle A_{l_n}, B_{l_n} \rangle \\
\hline
k_1, \langle A_{k_1}, B_{k_1} \rangle & \langle M_{k_1,l_1}, N_{k_1,l_1} \rangle & \ldots & \langle M_{k_1,l_j}, N_{k_1,l_j} \rangle & \ldots & \langle M_{k_1,l_n}, N_{k_1,l_n} \rangle \\
\vdots & \vdots & \ldots & \vdots & \ldots & \vdots \\
k_i, \langle A_{k_i}, B_{k_i} \rangle & \langle M_{k_i,l_1}, N_{k_i,l_1} \rangle & \ldots & \langle M_{k_i,l_j}, N_{k_i,l_j} \rangle & \ldots & \langle M_{k_i,l_n}, N_{k_i,l_n} \rangle \\
\vdots & \vdots & \ldots & \vdots & \ldots & \vdots \\
k_m, \langle A_{k_m}, B_{k_m} \rangle & \langle M_{k_m,l_1}, N_{k_m,l_1} \rangle & \ldots & \langle M_{k_m,l_j}, N_{k_m,l_j} \rangle & \ldots & \langle M_{k_m,l_n}, N_{k_m,l_n} \rangle
\end{array} ,
$$

where for every $1 \le i \le m$, $1 \le j \le n$:

$$M_{k_i,l_j}, N_{k_i,l_j} \subseteq [0, 1], \sup M_{k_i,l_j} + \sup N_{k_i,l_j} \leq 1,$$

$$A_{k_i}, B_{k_i} \subseteq [0, 1], \sup A_{k_i} + \sup B_{k_i} \leq 1, A_{l_j}, B_{l_j} \subseteq [0, 1], \sup A_{l_j} + \sup B_{l_j} \leq 1$$

and here and below,

$$K^* = \{\langle k_i, A_{k_i}, B_{k_i}\rangle | k_i \in K\} = \{\langle k_i, A_{k_i}, B_{k_i}\rangle | 1 \leq i \leq m\},$$

$$L^* = \{\langle l_j, A_{l_j}, B_{l_j}\rangle | l_j \in L\} = \{\langle l_j, A_{l_j}, B_{l_j}\rangle | 1 \leq j \leq n\}.$$

2.2.2 Basic Operations over EIVIFIMs

Let "$*$" and "$\circ$" be two fixed operations over IVIFPs and let

$$\langle M, N\rangle * \langle P, Q\rangle = \langle M *_l P, N *_r Q\rangle,$$

where the forms of operations "$*_l$" and "$*_r$" are determined by the form of the operation "$*$", as it is shown in the examples.

For the EIVIFIMs $A = [K^*, L^*, \{\langle M_{k_i,l_j}, N_{k_i,l_j}\rangle\}]$, $B = [P^*, Q^*, \{\langle R_{p_r,q_s}, S_{p_r,q_s}\rangle\}]$, with

$$K^* = \{\langle k_i, A_{k_i}, B_{k_i}\rangle | k_i \in K\}, L^* = \{\langle l_j, A_{l_j}, B_{l_j}\rangle | l_j \in L\},$$

$$P^* = \{\langle p_r, C_{p_r}, D_{q_s}\rangle | p_r \in P\}, Q^* = \{\langle q_s, C_{p_r}, D_{q_s}\rangle | q_s \in Q\}$$

operations that are analogous to the usual matrix operations of addition and multiplication are defined, as well as other specific ones.

Addition-($*$)

$$A \oplus_{(*)} B = [T^*, V^*, \{\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w}\rangle\}],$$

where

$$T^* = K^* \cup P^* = \{\langle t_u, A_{t_u}, B_{t_u}\rangle | t_u \in K \cup P\},$$

$$V^* = L^* \cup Q^* = \{\langle v_w, A_{v_w}, B_{v_w}\rangle | v_w \in L \cup Q\},$$

$$A_{t_u} = \begin{cases} A_{k_i}, & \text{if } t_u = k_i \in K - P \\ A_{p_r}, & \text{if } t_u = p_r \in P - K \\ A_{k_i} \cup A_{p_r}, & \text{if } t_u = k_i = p_r \in K \cap P \end{cases},$$

$$B_{v_w} = \begin{cases} B_{l_j}, & \text{if } v_w = l_j \in L - Q \\ B_{q_s}, & \text{if } t_w = q_s \in Q - L \\ B_{v_w} \cap B_{q_s}, & \text{if } t_w = v_w = q_s \in L \cap Q \end{cases},$$

and

$$\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w} \rangle = \begin{cases} \langle M_{k_i,l_j}, N_{k_i,l_j} \rangle, & \text{if } t_u = k_i \in K \\ & \quad \text{and } v_w = l_j \in L - Q \\ & \quad \text{or } t_u = k_i \in K - P \\ & \quad \text{and } v_w = l_j \in L; \\ \langle R_{p_r,q_s}, S_{p_r,q_s} \rangle, & \text{if } t_u = p_r \in P \\ & \quad \text{and } v_w = q_s \in Q - L \\ & \quad \text{or } t_u = p_r \in P - K \\ & \quad \text{and } v_w = q_s \in Q; \\ \langle M_{k_i,l_j} *_l R_{p_r,q_s}, & \text{if } t_u = k_i = p_r \in K \cap P \\ N_{k_i,l_j} *_r S_{p_r,q_s} \rangle, & \quad \text{and } v_w = l_j = q_s \in L \cap Q; \\ \langle [0,0], [1,1] \rangle, & \text{otherwise} \end{cases}$$

Termwise multiplication-(*)

$$A \otimes_{(*)} B = [T^*, V^*, \{\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w} \rangle\}],$$

where

$$T^* = K^* \cap P^* = \{\langle t_u, A_{t_u}, B_{t_u} \rangle | t_u \in K \cup P\},$$

$$V^* = L^* \cap Q^* = \{\langle v_w, A_{v_w}, B_{v_w} \rangle | v_w \in L \cup Q\},$$

$$A_{t_u} = A_{k_i} \cap A_{p_r}, \text{ for } t_u = k_i = p_r \in K \cap P,$$

$$B_{v_w} = B_{v_w} \cup B_{q_s}, \text{ for } v_w = l_j = q_s \in L \cap Q$$

and

$$\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w} \rangle = \langle M_{k_i,l_j} *_l R_{p_r,q_s}, N_{k_i,l_j} *_r S_{p_r,q_s} \rangle.$$

Multiplication-$(\circ, *)$

$$A \odot_{(\circ,*)} B = [T^*, V^*, \{\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w} \rangle\}],$$

where

$$T^* = (K \cup (P - L))^* = \{\langle t_u, A_{t_u}, B_{t_u} \rangle | t_u \in K \cup (P - L)\},$$

$$V^* = (Q \cup (L - P))^* = \{\langle v_w, A_{v_w}, B_{v_w}\rangle | v_w \in Q \cup (L - P)\},$$

$$A_{t_u} = \begin{cases} A_{i}^{k}, & \text{if } t_u = k_i \in K \\ A_{p_r}, & \text{if } t_u = p_r \in P - L \end{cases},$$

$$B_{v_w} = \begin{cases} B_{l_j}, & \text{if } v_w = l_j \in L - P \\ \\ B_{q_s}, & \text{if } t_w = q_s \in Q \end{cases},$$

and

$$\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w}\rangle =$$

$$= \begin{cases} \langle M_{k_i,l_j}, N_{k_i,l_j}\rangle, & \text{if } t_u = k_i \in K \\ & \quad \text{and } v_w = l_j \in L - P - Q; \\ \\ \langle R_{p_r,q_s}, S_{p_r,q_s}\rangle, & \text{if } t_u = p_r \in P - L - K \\ & \quad \text{and } v_w = q_s \in Q; \\ \\ \langle \underset{l_j = p_r \in L \cap P}{\circ_l} (M_{k_i,l_j} *_l R_{p_r,q_s})), & \text{if } t_u = k_i \in K \\ \underset{l_j = p_r \in L \cap P}{\circ_r} (N_{k_i,l_j} *_r S_{p_r,q_s})\rangle, & \quad \text{and } v_w = q_s \in Q; \\ \\ \langle [0, 0], [1, 1]\rangle, & \text{otherwise} \end{cases}$$

Structural subtraction

$$A \ominus B = [T^*, V^*, \{\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w}\rangle\}],$$

where

$$T^* = (K - P)^* = \{\langle t_u, A_{t_u}, B_{t_u}\rangle | t_u \in K - P\},$$

$$V^* = (L - Q)^* = \{\langle v_w, A_{v_w}, B_{v_w}\rangle | v_w \in L - Q\},$$

for the set–theoretic subtraction operation and

$$A_{t_u} = A_{k_i}, \text{ for } t_u = k_i \in K - P,$$

$$\beta_w^v = B_{l_j}, \text{ for } v_w = l_j \in L - Q$$

and

$$\langle \Phi_{t_u,v_w}, \Psi_{t_u,v_w}\rangle = \langle M_{k_i,l_j}, N_{k_i,l_j}\rangle, \text{ for } t_u = k_i \in K - P \text{ and } v_w = l_j \in L - Q.$$

Negation of an EIVIFIM

$$\neg A = [T^*, V^*, \{\neg\langle M_{k_i,l_j}, N_{k_i,l_j}\rangle\}],$$

where $\neg$ is one of the possible interval valued intuitionistic fuzzy negations that can be defined on the basis of the definitions of intuitionistic fuzzy negations from [3].

Termwise subtraction

$$A -_{\circ,*} B = A \oplus_{\circ,*} \neg B.$$

Transposition over an EIVIFIM: With A' let us denote the transposed IM of A.

Operations "reduction", "projection" and "substitution" are similar to the respective operations defined in Chap. 1, Sects. 1.6–1.8 of [4].

Reduction over an EIVIFIM: We use symbol "$\perp$" for lack of some component in the separate definitions. In some cases, it is suitable to change this symbol with "$\langle[0, 0], [1, 1]\rangle$".

The operations $(k, \perp)$- and $(\perp, l)$-reduction of a given IVIFIM A is defined by:

$$A_{(k,\perp)} = [(K - \{k\})^*, L^*, \{\langle P_{t_u,v_w}, Q_{t_u,v_w}\rangle\}]$$

where

$$\langle P_{t_u,v_w}, Q_{t_u,v_w}\rangle = \langle M_{k_i,l_j}, N_{k_i,l_j}\rangle$$

for $t_u = k_i \in K - \{k\}$ and $v_w = l_j \in L$ and

$$A_{(\perp,l)} = [K^*, (L - \{l\})^*, \{\langle P_{t_u,v_w}, Q_{t_u,v_w}\rangle\}],$$

where

$$\langle P_{t_u,v_w}, Q_{t_u,v_w}\rangle = \langle M_{k_i,l_j}, N_{k_i,l_j}\rangle$$

for $t_u = k_i \in K$ and $v_w = l_j \in L - \{l\}$.

Projection over an EIVIFIM: Let $M \subseteq K$ and $N \subseteq L$ and A be an EIVIFIM. Then,

$$pr_{M,N} A = [M, N, \{\langle P_{t_u,v_w}, Q_{t_u,v_w}\rangle\}],$$

where for each $k_i \in M$ and each $l_j \in N$,

$$\langle P_{k_i,l_j}, QN_{k_i,l_j}\rangle = \langle M_{k_i,l_j}, N_{k_i,l_j}\rangle.$$

Substitution over an EIVIFIM: Local substitution over an EIVIFIM A is defined for the couples of indices (p, k) and/or (q, l), respectively, by

$$\left[\frac{p}{k}; \perp\right] A = \left[(K^* - \{k\}) \cup \{p\}, L^*, \{a_{k,l}\}\right],$$

$$\left[\perp; \frac{q}{l}\right] A = \left[K^*, (L^* - \{l\}) \cup \{q\}, \{a_{k,l}\}\right].$$

Index type operations:
In [28, 29] are defined index type operations for IFIM A. Here, we extend some of them for IVIFIM A:

$$Index_{\{(\min/\max)/(\min_\square/\max_\square)/(\min_\diamond/\max_\diamond)(\min_R/\max_R)\}(\not\perp),k_i}(A)$$

$$= \{\langle k_i, l_{v_1}\rangle, \ldots, \langle k_i, l_{v_x}\rangle, \ldots, \langle k_i, l_{v_V}\rangle\},$$

where $\langle k_i, l_{v_x}\rangle$ (for $1 \le i \le m$, $1 \le v \le n$, $1 \le x \le V$) are the indices of the minimum/maximum IVIFP of the k_i-th row of A with no empty value in accordance with the relations (2).

$$AGIndex_{\{(\min_R/\max_R)\}(\notin F)(\not\perp)(\neq\langle[0,0],[1,1]\rangle)(M\neq[0,0])}(A)$$

$$= \{\langle k_{w_1}, l_{v_1}\rangle, \ldots, \langle k_{w_y}, l_{v_y}\rangle, \ldots, \langle k_{w_W}, l_{v_V}\rangle\}$$

which finds the indexes of the minimum/ maximum element between the elements of A, whose indexes respectively $\notin F$ or with no empty value, or not equal to $\langle[0, 0], [1, 1]\rangle$, or with non-zero degree of membership in accordance with the relations (2).

$$Index_{(\langle M,N\rangle)}(A) = \{\langle k_{w_1}, l_{v_1}\rangle, \ldots, \langle k_{w_y}, l_{v_y}\rangle, \ldots, \langle k_{w_W}, l_{v_V}\rangle\},$$

where $\langle k_{w_y}, l_{v_y}\rangle$ (for $1 \le i \le m$) are the indices of the elements equal to the IVIFP $\langle M, N\rangle$ of A.

$$AGIndex^1_{\{(\min_R/\max_R)\}(\notin F)/(\not\perp)/(\neq\langle[0,0],[1,1]\rangle)}(A)$$

$$= \{k_{w_1}, \ldots, k_{w_y} \ldots, \langle k_{w_W}\rangle\}$$

which finds the index of the minimum/ maximum element between the elements $\langle A_i^k, B_i^k\rangle$ of K^* (the first dimension of A), whose indexes respectively $\notin F$ or with no empty value, or not equal to $\langle[0, 0], [1, 1]\rangle$ in accordance with the relations (2).

$$AGIndex^2_{\{(\min_R/\max_R)\}(\notin F)/(\not\perp)/(\neq\langle[0,0],[1,1]\rangle)}(A) = \{l_{v_1}, \ldots, l_{v_y} \ldots, l_{v_V}\},$$

which finds the index of the minimum/ maximum element between the elements $\langle A_j^l, B_j^l\rangle$ of L^* (the second dimension of A), whose indexes respectively $\notin F$ or with no empty value, or not equal to $\langle[0, 0], [1, 1]\rangle$ in accordance with the relations (2).

Aggregation operations over EIVIFIMs

Let us have two IVIFPs $x = \langle[A, B]\rangle$ and $y = \langle[C, D]\rangle$.

We use following three operations $\#_q$, $(q \le i \le 3)$ for scaling aggregation evaluations:

$$x\#_1 y = \langle \big[min(\inf A, \inf C), min(\sup A, \sup C)\big],$$

$$\big[max(\inf B, \inf D), max(\sup B, \sup D)\big]\rangle;$$

$$x\#_2 y = \langle \big[average(\inf A, \inf C), average(\sup A, \sup C)\big],$$

$$\big[average(\inf B, \inf D), average(\sup B, \sup D)\big]\rangle;$$

$$x\#_3 y = \langle \big[max(\inf A, \inf C), max(\sup A, \sup C)\big],$$

$$\big[min(\inf B, \inf D), min(\sup B, \sup D)\big]\rangle;$$

From the theorem in [30] follows that: $x\#_1 y \leq x\#_2 y \leq x\#_3 y$.

Let the EIVIFIM A be given and let $k_0 \notin K$ and $l_0 \notin L$ be two fixed indices. In [31] is introduced the following aggregation operation over A by the dimension K

$$\alpha_{K,\#_q}(A, k_0)$$

$$= \begin{array}{c|ccc} & l_1, \langle A_1^l, B_1^l\rangle & \dots & l_n, \langle A_n^l, B_n^l\rangle \\ \hline k_0, \underset{i=1}{\overset{m}{\#_q}} \langle A_i^k, B_i^k\rangle & \underset{i=1}{\overset{m}{\#_q}} \langle M_{k_i,l_1}, N_{k_i,l_1}\rangle & \dots & \underset{i=1}{\overset{m}{\#_q}} \langle M_{k_i,l_n}, N_{k_i,l_n}\rangle \end{array},$$

where $1 \leq q \leq 3$.

In [7] are introduced the following two operations over A by the dimensions K and L:

$(*)$-row-aggregation

$$\rho_{(*)}(A, k_0) = \begin{array}{c|cccc} & & & & l_1, \langle A_{l_1}, B_{l_1}\rangle & \dots \\ \hline k_0, \langle & \underset{1\leq i\leq m}{*_l} A_{k_i}, & \underset{1\leq i\leq m}{*_r} B_{k_i}\rangle & \langle & \underset{1\leq i\leq m}{*_l} M_{k_i,l_1}, & \underset{1\leq i\leq m}{*_r} N_{k_i,l_1}\rangle \dots \end{array}$$

$$\begin{array}{c} \dots \quad l_n, \langle A_{l_n}, B_{l_n}\rangle \\ \hline \dots \langle \underset{1\leq i\leq m}{*_l} M_{k_i,l_n}, \underset{1\leq i\leq m}{*_r} N_{k_i,l_n}\rangle \end{array},$$

$(*)$-column-aggregation

$$\sigma_*(A, l_0) = \begin{array}{c|c} & l_0, \langle \underset{1\le i\le m}{*_l} A_{l_j}, \underset{1\le i\le m}{*_r} B_{l_j} \rangle \\ \hline k_1, \langle A_{k_1}, B_{k_1} \rangle & \langle \underset{1\le i\le m}{*_l} M_{k_1,l_j}, \underset{1\le i\le m}{*_r} N_{k_1,l_j} \rangle \\ \vdots & \vdots \\ k_m, \langle A_{k_m}, B_{k_m} \rangle & \langle \underset{1\le i\le m}{*_l} M_{k_m,l_j}, \underset{1\le i\le m}{*_r} N_{k_m,l_j} \rangle \end{array}.$$

Aggregate global internal operation:

$$AGIO_{\oplus_{(\max,\min)}}(A).$$

This operation [28] finds the "$\oplus_{(\max,\min)}$"-operation of all the matrix elements.

Internal subtraction of IMs' components [27, 28, 31]:

$$IO_{-(\max,\min)}(\langle k_i, l_j, A\rangle, \langle p_r, q_s, B\rangle) = [K, L, \{\langle \gamma_{t_u,v_w}, \delta_{t_u,v_w}\rangle\}]$$

$$\langle \gamma_{t_u,v_w}, \delta_{t_u,v_w}\rangle$$

$$= \begin{cases} \langle M_{t_u,v_w}, N_{t_u,v_w}\rangle, & \text{if } t_u \ne k_i \in K, v_w \ne l_j \in L, \\ \langle [\max(0, \min(\inf M - \sup P, 1 - \sup N + \sup Q)), & \text{if } t_u = k_i \in K,\ v_w = l_j \in L, \\ \max(0, \min(\sup M - \inf P, 1 - \sup N + \sup Q))], & \\ [\min(1, \min(\inf N + \inf Q, 1 - \sup M + \inf P)), & \\ \min(1, \sup N + \sup Q, 1 - \sup M + \inf P)]\rangle. & \end{cases}$$

where $k_i \in K,\ l_j \in L;\ p_r \in P,\ q_s \in Q$.

2.2.3 Relations Over EIVIFIMs

Let the two EIVIFIMs $A = [K^*, L^*, \{\langle A_{k,l}, B_{k,l}\rangle\}]$ and $B = [P^*, Q^*, \{\langle C_{p,q}, D_{p,q}\rangle\}]$ be given. We shall introduce the following definitions where $\subset$ and $\subseteq$ denote the relations *"strong inclusion"* and *"weak inclusion"*.

The strict relation "inclusion about dimension" is

$$A \subset_d B \text{ iff } (((K^* \subset P^*) \ \&\ (L^* \subset Q^*)) \ \vee\ ((K^* \subseteq P^*) \ \&\ (L^* \subset Q^*))$$

$$\vee((K^* \subset P^*) \ \&\ (L^* \subseteq Q^*))) \ \&\ (\forall k \in K)(\forall l \in L)(\langle A_{k,l}, B_{k,l}\rangle = \langle C_{k,l}, D_{k,l}\rangle).$$

The non-strict relation "inclusion about dimension" is

$$A \subseteq_d B \text{ iff } (K^* \subseteq P^*) \ \& \ (L^* \subseteq Q^*) \ \& \ (\forall k \in K)(\forall l \in L)$$

$$(\langle A_{k,l}, B_{k,l}\rangle = \langle C_{k,l}, D_{k,l}\rangle).$$

The strict relation "inclusion about value" is

$$A \subset_v B \text{ iff } (K^* = P^*) \ \& \ (L^* = Q^*) \ \& \ (\forall k \in K)(\forall l \in L)$$

$$(\langle A_{k,l}, B_{k,l}\rangle \subset^* \langle C_{k,l}, D_{k,l}\rangle).$$

The non-strict relation "inclusion about value" is

$$A \subseteq_v B \text{ iff } (K^* = P^*) \ \& \ (L^* = Q^*) \ \& \ (\forall k \in K)(\forall l \in L)$$

$$(\langle A_{k,l}, B_{k,l}\rangle \subseteq^* \langle C_{k,l}, D_{k,l}\rangle).$$

The strict relation "inclusion" is

$$A \subset_* B \text{ iff } (((K^* \subset P^*) \ \& \ (L^* \subset Q^*)) \ \vee \ ((K^* \subseteq P^*) \ \& \ (L^* \subset Q^*))$$

$$\vee \ ((K^* \subset P^*) \ \& \ (L^* \subseteq Q^*))) \ \& \ (\forall k \in K)(\forall l \in L)(\langle A_{k,l}, B_{k,l}\rangle \subset^* \langle C_{k,l}, D_{k,l}\rangle).$$

The non-strict relation "inclusion" is

$$A \subseteq_* B \text{ iff } (K^* \subseteq P^*) \ \& \ (L^* \subseteq Q^*) \ \& \ (\forall k \in K)(\forall l \in L)$$

$$(\langle A_{k,l}, B_{k,l}\rangle \subseteq^* \langle C_{k,l}, D_{k,l}\rangle).$$

2.3 Interval-Valued Intuitionistic Fuzzy Graphs (IVIFGs)

First, we remind six versions of operation over IVIFSs, namely Cartesian products of two IVIFSs. Let A be an IVIFS over E_1 and B—over E_2. In [6] are defined six versions of the Cartesian products of two IVIFSs:

$$
\begin{aligned}
A \times_1 B = \{\langle\langle x, y\rangle, [&\inf M_A(x).\inf M_B(y), \sup M_A(x).\sup M_B(y)],\\
&[\inf N_A(x).\inf N_B(y), \sup N_A(x).\sup N_B(y)]\rangle \mid x \in E_1, y \in E_2\}\\
A \times_2 B = \{\langle\langle x, y\rangle, [&\inf M_A(x) + \inf M_B(y) - \inf M_A(x).\inf M_B(y),\\
&\sup M_A(x) + \sup M_B(y) - \sup M_A(x).\sup M_B(y)],\\
&[\inf N_A(x).\inf N_B(y), \sup N_A(x).\sup N_B(y)]\rangle \mid x \in E_1, y \in E_2\}\\
A \times_3 B = \{\langle\langle x, y\rangle, [&\inf M_A(x).\inf M_B(y), \sup M_A(x).\sup M_B(y)],\\
&[\inf N_A(x) + \inf N_B(y) - \inf N_A(x).\inf N_B(y),\\
&\sup N_A(x) + \sup N_B(y) - \sup N_A(x).\sup N_B(y)]\rangle \mid x \in E_1, y \in E_2\}\\
A \times_4 B = \{\langle\langle x, y\rangle, [&\min(\inf M_A(x), \inf M_B(y)), \min(\sup M_A(x), \sup M_B(y))],\\
&[\max(\inf N_A(x), \inf N_B(y)), \max(\sup N_A(x), \sup N_B(y))]\rangle \mid x \in E_1, y \in E_2\}\\
A \times_5 B = \{\langle\langle x, y\rangle, [&\max(\inf M_A(x), \inf M_B(y)), \max(\sup M_A(x), \sup_B(y))],\\
&[\min(\inf N_A(x), \inf N_B(y)), \min(\sup N_A(x), \sup N_B(y))]\rangle \mid x \in E_1, y \in E_2\}\\
A \times_6 B = \{\langle\langle x, y\rangle, [&\tfrac{\inf M_A(x)+\inf M_B(y)}{2}, \tfrac{\sup M_A(x)+\sup M_B(y)}{2}],\\
&[\tfrac{\inf N_A(x)+\inf N_B(y)}{2}, \tfrac{\sup N_A(x)+\sup N_B(y)}{2}]\rangle \mid x \in E_1, y \in E_2\}
\end{aligned}
\tag{3}
$$

The concept of the intuitionistic fuzzy graph was introduced in 1994 in [23]. In [6] are proposed new definitions of IVIFGs. Let us present one of them.

Let us have a fixed set of vertices $\mathscr{V} = \{v_1, v_2, \ldots, v_n\}$. An $(\circ)$-**(IVIFS, IVIFS)-IFG** G (over $\mathscr{V}$) will be the ordered pair $G = (V^*, A^*)$, where

$$V \subset \mathscr{V}, V^* = \{\langle v, M_V(v), N_V(v)\rangle | v \in V\},$$

$$A \subset V \times V, A^* = \{\langle\langle x, y\rangle, M_A(x, y), N_A(x, y)\rangle | \langle x, y\rangle \in V \times V\}$$

and functions $M_V : \mathscr{V} \to \mathscr{P}([0, 1])$ and $N_V : \mathscr{V} \to \mathscr{P}([0, 1])$ define the degree of membership and the degree of non-membership, respectively, of the element $v \in \mathscr{V}$ to the set V; functions $M_A : E_1 \times E_2 \to \mathscr{P}([0, 1])$ and $N_A : E_1 \times E_2 \to \mathscr{P}([0, 1])$ define the degree of membership and the degree of non-membership, respectively, of the element $\langle x, y\rangle \in E_1 \times E_2$ to the set $A \subseteq E_1 \times E_2$; these functions have the forms of the corresponding components of the $\circ$-Cartesian product over IVIFSs, where $\circ \in \{\times_1, \ldots, \times_6\}$ and for all $\langle x, y\rangle \in E_1 \times E_2$,

$$0 \leq \sup M_V(x) + \sup N_V(x) \leq 1,$$

$$0 \leq \sup M_A(x, y) + \sup N_A(x, y) \leq 1.$$

The parameters of the IVIFG G are IVIFPs. The expert approach described in detail in [3, 8] may be used to determine the distances between any two vertices and the existence of the vertices of the graph in the form of IVIFPs. Each expert is asked

to evaluate the distances respect to each predefined criterion and the existence of a vertex. Due to several uncertainties, the expert is not certain about the distances (or existence of a vertex). Also due to some uncontrollable factors, the expert hesitates in prediction of the distances between any two vertices. According to the past experience of the expert the distances are estimated as interval-valued intuitionistic fuzzy numbers after a thorough discussion, interpreted in the interval-valued intuitionistic fuzzy framework: these numbers express a "positive" and a "negative" evaluations, respectively.

Now, for the graph $G = (V, A)$ was constructed the Extended Interval-Valued Intuitionistic Fuzzy Graph (EIVIFG) $G^* = (V^*, A^*)$ [6]. It has the following IM-representation as adjacency EIVIFIM C:

$$[V^*, V^*, \{\langle M_{v_i,v_j}, N_{v_i,v_j}\rangle\}] = \begin{array}{c|ccc} & v_1, \langle A_{v_1}, B_{v_1}\rangle & \dots & v_n, \langle A_{v_n}, B_{v_n}\rangle \\ \hline v_1, \langle A_{v_1}, B_{v_1}\rangle & \langle M_{v_1,v_1}, N_{v_1,v_1}\rangle & \dots & \langle M_{v_1,v_n}, N_{v_1,v_n}\rangle \\ \vdots & \vdots & \dots & \vdots \\ v_i, \langle A_{v_i}, B_{v_i}\rangle & \langle M_{v_i,v_1}, N_{v_i,v_1}\rangle & \dots & \langle M_{v_i,v_n}, N_{v_i,v_n}\rangle \\ \vdots & \vdots & \dots & \vdots \\ v_n, \langle A_{v_n}, B_{v_n}\rangle & \langle M_{v_n,v_1}, N_{v_n,v_1}\rangle & \dots & \langle M_{v_n,v_n}, N_{v_n,v_n}\rangle \end{array}, \tag{4}$$

where n is the cardinality of set V and for every $1 \le i \le n, 1 \le j \le n$: $C_{v_i,v_j} = \langle M_{v_i,v_j}, N_{v_i,v_j}\rangle$ and $\langle A_{v_i}, B_{v_i}\rangle$ are IVIFPs. As in the ordinary case, the vertex $v_p \in V$ has a loop if and only if $C_{p,p} = \langle M_{p,p}, N_{p,p}\rangle$ for the vertex v_p and $M_{p,p} \neq [0, 0]$ (hence $N_{p,p} \neq [1, 1]$).

A path in an IVIFG is a sequence of vertices $u_1, u_2, \dots, u_n$ such that $M_{u_{i-1},u_i} \neq [0, 0]$, $1 \le i \le n$. An interval-valued intuitionistic fuzzy Hamiltonian path is a path that passes through each of the vertices in an IVIFG exactly once. An interval-valued intuitionistic fuzzy Hamiltonian cycle is a cycle that visits every vertex in an IVIFG once with no repeats and it must start and end at the same vertex.

Proposition 1 *In an IVIFG, if every vertex has exactly two adjacent vertices, then there exists a Hamiltonian cycle.*

The proof of the proposition is analogous to that of [15].

3 Algorithms for Hamiltonian Cycle in an IVIFG

Let us be given IVIFG $G^* = (V^*, A^*)$ with the IM-representation as EIVIFIM C with a structure (4). The purpose is to find the Hamiltonian cycle in G^*. Let us extend the algorithms for Hamiltonian cycle in an IFG from [29] to interval-valued intuitionistic fuzzy ones, based on IVIFSs and IMs concepts.

Algorithm 1: Minimum edge degree algorithm to find interval-valued intuitionistic fuzzy Hamiltonian cycle

Let us define

$$X[V^*, V^*, \{\langle \rho(v_i, v_j), \sigma(v_i, v_j)\rangle\}] = \begin{array}{c|ccc} & v_1, \langle A_{v_1}, B_{v_1}\rangle & \dots & v_n, \langle A_{v_n}, B_{v_n}\rangle \\ \hline v_1, \langle A_{v_1}, B_{v_1}\rangle & \langle \rho_{v_1,v_1}, \sigma_{v_1,v_1}\rangle & \dots & \langle \rho_{v_1,v_n}, \sigma_{v_1,v_n}\rangle \\ \vdots & \vdots & \dots & \vdots \\ v_i, \langle A_{v_i}, B_{v_i}\rangle & \langle \rho_{v_i,v_1}, \sigma_{v_i,v_1}\rangle & \dots & \langle \rho_{v_i,v_n}, \sigma_{v_i,v_n}\rangle \\ \vdots & \vdots & \dots & \vdots \\ v_n, \langle A_{v_n}, B_{v_n}\rangle & \langle \rho_{v_n,v_1}, \sigma_{v_n,v_1}\rangle & \dots & \langle \rho_{v_n,v_n}, \sigma_{v_n,v_n}\rangle \end{array},$$

where for i and $j (1 \le i \le n, 1 \le j \le n)$: x_{v_i,v_j} and $\langle A_{v_i}, B_{v_i}\rangle$ are IVIFPs.

Let us we create the following auxiliary IMs:
(1) $S = [V^*, V^*, \{s_{v_i,v_j}\}]$, such that $S = C$ i.e. $c_{v_i,v_j} = s_{v_i,v_j}$ $\forall v_i \in V^*, \forall v_j \in V^*$);
(2)

$$RC[V^*, e_0] = \begin{array}{c|c} & e_0 \\ \hline v_1 & rc_{v_1,e_0} \\ \vdots & \vdots \\ v_i & rc_{v_i,e_0} \\ \vdots & \vdots \\ v_n & rc_{v_n,e_0} \end{array},$$

where for $1 \le i \le n$: $rc_{v_i,e_0} = \{0, 1\}$ depending on whether the v_i-th vertex of the matrix S is crossed out (introduced in the Hamiltonian path) or not. When the algorithm starts, $rc_{k_i,e_0} = 0$, $x_{v_i,v_j} = \langle [0, 0][1, 1]\rangle$ $(\forall v_i \in V*, \forall v_j \in V*)$. With $|A|$ let us we denote the number of elements of the A, where A is an IM.

We will describe the algorithm for determining a Hamiltonian path in G^*, interpreted with the tools of EIVIFIMs and interval-valued intuitionistic fuzzy logic extending the algorithm from [29]:

Step 1. Construct the EIVIFIM C for the given IVIFG G^* and create EIVIFIM S such that $S := C$;

Then check the condition of the proposition 1. for the existence of a Hamiltonian cycle in G^* :

for $i = 1$ to n then

$$\{D = AGIndex_{\{(\min_R / \max_R)\}(M \neq [0,0])}(S) = \{\langle k_{w_i}, l_{v_1}\rangle, \dots, \langle k_{w_i}, l_{v_y}\rangle, \dots, \langle k_{w_i}, l_{v_V}\rangle\}\}.$$

If $|D| < 2$ then {the Hamiltonian path does not exist and the algorithm **Stop**}
else Go to *Step 2*}.

Step 2. Search for a minimum IVIFP in the S with membership degree $M \neq [0, 0]$ in accordance with the relations (2). If there are several such elements, then we choose any one.

$$AGIndex_{\{(\min_R)\}(M \neq [0,0])}(S) = \langle k_z, l_z\rangle (1 \le z \le n).$$

If $x_{k_z,l_z} = \langle[0,0][1,1]\rangle$,
then
{
$rc[k_z, e_0] = 1$; $S_{(k_z,\perp)}$; $x_{k_z,l_z} = \langle[1,1][0,0]\rangle$
};
Go to *Step 3*.
Step 3. If the minimum IVIFP s_{k_z,l_z} in the IM S does not allow for a fuzzy Hamiltonian path, then the next higher minimum IVIFP is selected.

$$AGIndex_{\{(\min_R)\}(M\neq[0,0])}(S) = \langle k_x, l_x\rangle (1 \leq x \leq n);$$

Go to *Step 4*.
Step 4. Identify the k_x-th row and l_x-th column in the S, where the minimum IVIFP appears. We add to Hamiltonian the vertex l_x i.e. from k_x it reaches l_x. $rc_{k_x,e_0} = rc_{l_x,e_0} = 1$, $x_{k_x,l_x} = \langle[1,1],[0,0]\rangle$ and S is reduced by $S_{(\perp,l_x)}$; Go to *Step 5*.
Step 5. Search for a minimum IVIFP of the l_x-th row, such that: it forms an interval-valued intuitionistic fuzzy Hamiltonian path; if the minimum value occurs more than once, then an IVIFP is selected for a Hamiltonian path.

The operation is:

$$AGIndex_{\{(\min_R)\}(M\neq[0,0])}\left(pr_{l_x,V^*}S\right) = \langle l_x, l_u\rangle;$$

Then $rc_{l_x,e_0} = rc_{l_u,e_0} = 1$, $x_{k_x,l_u} = \langle[1,1],[0,0]\rangle$ and the IM S is reduced by $S_{(l_x,l_u)}$.
Step 6. Repeat *Step 3* through *Step 4* row-wise until $|Index_{(1)}(RC)| = n$. With $|A|$ let us we denote the number of elements of the A, where A is an IM.
If $|Index_{(1)}(RC)| = n$ then
{
an interval-valued intuitionistic fuzzy Hamiltonian path with all n vertices of G^* is found and go to *Step 7*,
}
else
{
there is no interval-valued intuitionistic fuzzy Hamiltonian path and repeat *Step 2* or *Step 3* as required.
}
Step 7. If interval-valued intuitionistic fuzzy Hamiltonian path exists, then only one row k_l will be left out in the IM S. Select an IVIFP with degree of membership $M \neq [0,0]$ from that row to form an interval-valued intuitionistic fuzzy Hamiltonian cycle, if exists.
If we can select the IVIFP s_{k_l,k_z} with degree of membership $M \neq [0,0]$ then {the Hamiltonian cycle exists, $x_{k_l,k_z} = \langle[1,1],[0,0]\rangle$ and go to *Step 8*}
else the algoritm **Stop**.
Step 8. If $|Index_{(\langle[1,1],[0,0]\rangle)}(X)| = n$, then the optimal Hamiltonian path is obtained with minimum interval-valued intuitionistic fuzzy length according to (2). The IF length of the path is:

$$AGIO_{1_{\oplus_{(\max,\min)}}}\left(C \otimes_{(\min,\max)} X\right) \text{ or } AGIO_{2_{\oplus_{(\vee_2)}}}\left(C \otimes_{(\wedge_2)} X\right),$$

where $\vee_2$ and $\wedge_2$ are the operations from (1).

For an IVIFG with n vertices the algorithm visits all the permutations of the vertices, so the complexity is $O(n!)$.

Algorithm 2 Minimum vertex degree algorithm to find an interval-valued intuitionistic fuzzy Hamiltonian cycle

Step 1. We select a vertex $v_{min,i}$, whose $\langle A_{v_{min,i}}, B_{v_{min,i}}\rangle$ is the minimum IVIFPs $\langle A_{v_i}, B_{v_i}\rangle (1 \leq i \leq n)$. (If there is more than one vertex with same IVIFP, then choose any one).

Step 2. Select a vertex with IVIFP whose is next higher to the minimum IVIFP $\langle A_{v_{min,i}}, B_{v_{min,i}}\rangle$.

Step 3. Identify the adjacent vertices of the vertex with minimum IVIFP selected.

Step 4. We select the unvisited adjacent vertex of the minimum IVIFP $\langle A_{v_i}, B_{v_i}\rangle (1 \leq i \leq n)$. (If more than one adjacent vertex has the same minimum IVIFP, then choose any one vertex).

Step 5. *Step 3–Step 4* are repeated until an interval-valued IF Hamiltonian cycle is found else go to *Step 1*. or *Step 2*. as required.

This **algorithm 2** can be presented with similar IMs operations to that of the **algorithm 1**, but the operation $AGIndex^1_{\{(\min_R / \max_R)\}(M \neq [0,0])}(S)$ will be used instead $AGIndex_{\{(\min_R / \max_R)\}(M \neq [0,0])}(S)$.

In the algorithm 2, *step 3* and *step 4* need to be repeated starting with each vertex of IVIFG G^* to find all possible IVIF Hamiltonian cycles. We can identify the minimum length of IVIF Hamiltonian cycle(s).

4 An Example for Hamiltonian Cycle in IVIFG

The part of the airline network of the Wizz air [32] is modeled by an IVIFG $G^* = (V^*, A^*)$. The vertices of IVIFG are {Sofia airport (So), Dortmund airport (Do), Viena airport (Vi), Brussels South Charleroi Airport (Br), Barcelona airport (Ba)}.

The IVIFG has the following IM-representation as EIVIFIM $C[V^*, V^*]$:

	So, ⟨[0.5, 0.6], [0.1, 0.2]⟩	Do, ⟨[0.65, 0.75], [0.1, 0.2]⟩	…
So, ⟨[0.5, 0.6], [0.10, 0.20]⟩	⟨[0, 0], [1, 1]⟩	⟨[0.7, 0.80], [0.08, 0.18]⟩	…
Do, ⟨[0.65, 0.75], [0.1, 0.2]⟩	⟨[0.7, 0.8]; [0.08, 0.18]⟩	⟨[0, 0], [1, 1]⟩	…
Vi, ⟨[0.6, 0.7], [0.0, 0.1]⟩	⟨[0.3, 0, 4], [0.4, 0.5]⟩	⟨[0.25, 0, 35], [0.5, 0.6]⟩	…
Br, ⟨[0.7, 0.8], [0.0, 0.1]⟩	⟨[0.7, 0, 8], [0, 0.1]⟩	⟨[0.05, 0.1], [0.8, 0.9]⟩	…
Ba, ⟨[0.73, 0.83], [0.05, 0.15]⟩	⟨[0.8, 0.9], [0, 0.05⟩	⟨[0.4, 0.5], [0.38, 0.48]⟩	…

... Vi, $\langle[0.6, 0.7], [0, 0.1]\rangle$	Br, $\langle[0.7, 0.8], [0, 0.10]\rangle$	Ba, $\langle[0.73, 0.83], [0.05, 0.15]\rangle$
... $\langle[0.3, 0.4], [0.4, 0.5]\rangle$	$\langle[0.7, 0.8], [0, 0.1]\rangle$	$\langle[0.8, 0.9], [0, 0.05]\rangle$
... $\langle[0.25, 0.35], [0.5, 0.6]\rangle$	$\langle[0.05, 0.1], [0.8, 0.9]\rangle$	$\langle[0.4, 0.5], [0.38, 0.48]\rangle$
... $\langle[0, 0], [1, 1]\rangle$	$\langle[0.28, 0.38], [0.5, 0.6]\rangle$	$\langle[0.65, 0.75], [0.1, 0.2]\rangle$
... $\langle[0.28, 0.38], [0.5, 0.6]\rangle$	$\langle[0.0], [1, 1]\rangle$	$\langle[0.35, 0.45], [0.4, 0.5]\rangle$
... $\langle[0.65, 0.75], [0.1, 0.2]\rangle$	$\langle[0.35, 0.45], [0.4, 0.5]\rangle$	$\langle[0, 0], [1, 1]\rangle$

The IVIFPs, presented the vertices and the edges of G^* are calculated using the expert approach [8]. Each of the experts is asked to evaluate the interval of the degree of membership of IVIFP, corresponding to every vertex using the ratio of the air distance to all destinations from it to the total air distance of all the air roads. The minimum interval of the degree of membership proposed by the experts for the respective vertex forms its interval of the degree of membership.

The interval of the degree of non-membership of the vertex is calculted as

$$[[1, 1] - \text{ the maximum interval of the degree of membership}],$$

proposed by the experts. The approach to calculating the IVIFP for the length of each edge is similar. Each expert estimates the degree of membership of IVIFP, corresponding to every edge using the ratio of the air distance between two cities to the total air distance of all the air roads in the map of Wizz air.

Let us find a Hamiltonian cycle in the G^* using the algorithm 1:

Step 1. Let us create EIVIFIM C for the given IVIFG G^*. Then we create IM S such that $S := C$. The condition of the proposition 1 for the existence of a Hamiltonian cycle in G^* is met.

Step 2. The minimum IVIFP in the S with membersheep degree $M \neq [0, 0]$ is $s_{Do,Br} = \langle[0.05, 0.10], [0.8, 0.9]\rangle$.

Step 3. The minimum IVIFP in the S represents that from Dortmund airport it reaches Brussels South Charleroi Airport.

Step 4. In the row "Br" the minimum IVIFP is equal to $\langle[0.28, 0.38], [0.5, 0.6]\rangle = s_{Vi,Br}$. Therefore from Brussels South Charleroi it goes to Airport Vienna airport.

Step 5. Repeat *Steps 3–4*, the path obtained is: Do-Br-Vi-Ba-So.

Step 6. From the row "So", select the element $s_{So,Do} = \langle[0.7, 0.8], [0.08, 0.18]\rangle$ with membership degree $\neq [0, 0]$ to get an IVIF Hamiltonian cycle "Do-Br-Vi-Ba-So-Do." The IVIF length of the path is:

$$AGIO_{1_{\oplus_{(\max,\min)}}}\left(C \otimes_{(\min,\max)} X\right) = \langle[0.95, 0.99], [0, 0.005]\rangle.$$

The interval-valued degree of membership (acceptance) of this optimal solution is equal to $[0.95, 0.9]$ and its interval-valued degree of non-membership (non-acceptance) is equal to $[0, 0.005]$.

The example illustrates the reliability of the proposed IVIF approach.

5 Conclusion

In this paper it was developed an index-matrix interpretation to find IVIF Hamiltonian cycle in an IVIFG. The main contribution of our approach lies in its proposition of two index-matrix algorithms to find a Hamiltonian cycle in IVIFG, based on respectively on the minimum edge degree and the minimum vertex degree under the form of IVIFP, to handle with the uncertainty in the environment. The efficiency of the first algorithm was demonstrated by a real data example from the selected network map of Wizz air. In the paper also was defined three index type operations over EIVIFIMs. In the future, we will extend this algorithm for an application to the three-dimensional intuitionistic fuzzy intuitionistic fuzzy graphs [4].

References

1. Atanassov, K.T.: Intuitionistic fuzzy sets, VII ITKR Session, Sofia, 20-23 June 1983 (Deposed in Centr. Sci.-Techn. Library of the Bulg. Acad. of Sci., 1697/84) (in Bulgarian). Reprinted: Int. J. Bioautomation, **20** (S1), S1–S6 (2016)
2. Atanassov, K.: Generalized index matrices. Comptes rendus de l'Academie Bulgare des Sci **40**(11), 15–18 (1987)
3. Atanassov, K.: On intuitionistic fuzzy sets theory. STUDFUZZ, vol. 283. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29127-2
4. Atanassov, K.: Index matrices: towards an augmented matrix calculus. Stud. Comput. Intell. **573**. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10945-9
5. Atanassov, K.: Interval-valued intuitionistic fuzzy sets. Stud. Fuzziness Soft Comput. **388** (2020)
6. Atanassov, K.: Interval-valued intuitionistic fuzzy graphs. Notes IFSs **25**(1), 21–31 (2019)
7. Atanassov, K.: Extended interval valued intuitionistic fuzzy index matrices. In: Atanassov, K., et al. (eds.) Uncertainty and Imprecision in Decision Making and Decision Support: New Challenges, Solutions and Perspectives. IWIFSGN 2018. Advances in Intelligent Systems and Computing, vol. 1081. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-47024-1_1
8. Atanassov, K., Gargov, G.: Interval valued intuitionistic fuzzy sets. Fuzzy Sets Syst. **31**(3), 343–349 (1989)
9. Atanassov, K., Vassilev, P., Kacprzyk, J., Szmidt, E.: On interval valued intuitionistic fuzzy pairs. J. Univers. Math. **1**(3), 261–268 (2018)
10. Chvatal, V.: On Hamilton's ideals. J. Comb. Theory Ser B. **12**, 63–168 (1972)
11. Dirac, G.A.: Some theorems on abstract graphs. Proc. Lond. Math. Soc. **3**(1), 69–81 (1952)
12. Dudek, A., Ferrara, M.: Extensions of results on rainbow Hamilton cycles in uniform hypergraphs. Graphs Comb. **31**(3), 577–583 (2015)
13. Euler, L.: Solutio problematis ad geometriam situs pertinentis. Comment. Acad. Sci. Imp. Petropolitanae **8**, 128–140 (1736)
14. Fan, G.: New sufficient conditions for cycles in graphs. J. Comb. Theory Ser. B. **37**, 221–227 (1984)
15. Gani, A., Latha, S.: A new algorithm to find fuzzy Hamilton cycle in a fuzzy network using adjacency matrix and minimum vertex degree. SpringerPlus, Int. J. Pure Appl. Math. **5**, 1854 (2016)
16. Kauffman, A.: Introduction a la Theorie des Sous-emsembles Flous. Masson et Cie Editeurs, Paris (1973)
17. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Complexity of computations, pp. 85–103. Plemem Press, New York (1972)

18. Lee, C., Sudakov, B.: Dirac's theorem for random graphs. Rand Struct. Algorithm **41**, 293–305 (2012)
19. Mordeson, J.N., Nai, P.S.: Cycles and co-cycles of fuzzy graphs. Inf. Sci. **90**, 39–49 (1996)
20. Ore, O.: Note on Hamiltonian circuits. Am. Math. Mon. **67**, 55 (1960)
21. Rosenfeld, A.: Fuzzy graphs. In: Zadeh, L.A., Fu, K.S., Shimura, M. (eds.) Fuzzy Sets and Their Applications, pp. 77–95. Academic Press, New York (1975)
22. Shang, Y.: On the Hamiltonicity of random bipartite graphs. Indian J. Pure Appl. Math. **46**(2), 163–173 (2015)
23. Shannon, A., Atanassov, K.: A first step to a theory of the intuitionistic fuzzy graph. In: lakov, D. (ed.) Proceedings of the First Workshop on Fuzzy Based Expert Systems, pp. 59–61. Sofia (1994)
24. Skiena, S.: Hamiltonian cycles. Implementing discrete mathematics: combinatorics and graph theory with mathematica reading. Addison Wesley, New York, pp. 196–198 (1990)
25. Sovan, S., Sarkar, B.: Generalized fuzzy Euler graphs and generalized fuzzy Hamiltonian graphs. J. Intell. Fuzzy Syst. 3413–3419 (2018)
26. Szmidt, E., Kacprzyk, J.: Amount of information and its reliability in the ranking of Atanassov's intuitionistic fuzzy alternatives. In: Rakus-Andersson, E., Yager, R., Ichalkaranje, N., Jain, L.C. (eds.). Recent Advances in Decision Making, SCI, Springer, Heidelberg, vol. 222, pp. 7–19 (2009). https://doi.org/10.1007/978-3-642-02187-9_2
27. Traneva, V.: Internal operations over 3-dimensional extended index matrices. Proc. Jangjeon Math. Soc. **18**(4), 547–569 (2015)
28. Traneva, V., Tranev, S.: Index matrices as a tool for managerial decision making. Publ, House of the Union of Scientists, Bulgaria (2017). (in Bulgarian)
29. Traneva V., Tranev S.: Intuitionistic fuzzy hamiltonian cycle by index matrices. In: Proceedings of the 15th Conference on Computer Science and Information Systems (FedCSIS). Sofia, Bulgaria, pp. 345–348 (2020). https://doi.org/10.15439/2020F61
30. Traneva, V., Tranev, S., Stoenchev, M., Atanassov, K.: Scaled aggregation operations over two- and three-dimensional index matrices. Soft Comput. **22**, 5115–5120 (2019). https://doi.org/10.1007/00500-018-3315-6
31. Traneva, V., Atanassova, V., Tranev, S.: Three-dimensional interval valued intuitionistic fuzzy appointment model. In: Fidanova, S. (ed.) Recent Advances in Computational Optimization. Studies in Computational Intelligence, vol. 838, pp. 181–199. Springer, Cham. https://doi.org/10.1007/978-3-030-22723-4_12
32. http://wizz.air-bg.com. Last accessed 1 Feb 2021
33. Yeh, R.T., Bang, S.Y.: Fuzzy relations, fuzzy graphs and their applications to clustering analysis fuzzy sets and their applications to cognitive and decision processes. Academic Press, London, pp. 125–149 (1975)
34. Zadeh, L.: Fuzzy sets. Inf. Control **8**(3), 338–353 (1965)
35. Zhao, K., Hong-Jian, L., Shao, Y.: New sufficient condition for Hamiltonian graphs. Appl. Math. Let. **20**, 116–122 (2007)

On the Conceptual Optimization of Generalized Net Models

Velin Andonov, Stoyan Poryazov, and Emiliya Saranova

Abstract The problem of conceptual optimization of Generalized Nets (GNs) models is discussed. An overview of some operators for complexity of GNs and relations with respect to them is presented. Some new operators and relations are defined. The proposed approach to the conceptual optimization of GN models is applied to a GN model of a queuing system with finite capacity of the buffer and server, and FIFO discipline of service of the requests.

Keywords Generalized nets · Conceptual modeling · Conceptual optimization · Queuing system

1 Introduction

One of the first attempts to propose a definition of a set of quantifiable characteristics of a conceptual model; a measurement of the characteristics together with a fixed measurement of the decision-maker's preferences are done in Oren [9]. The proposed characteristics are: (1) size, (2) change pr. month, (3) data description inaccuracy, (4) semantic relevance, (5) semantic inaccuracy and (6) l/0-model size. Oren mentions that the conceptual model characteristics may be quantified absolutely or relatively and discusses the accuracy of quantifying the characteristics. Examples are not given.

Another approach can be found in [13]: Conceptual modelling is about abstracting a model that is fit-for-purpose and by this we mean a model that is (1) valid,

V. Andonov (✉) · S. Poryazov · E. Saranova
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Acad. G. Bonchev Str., Block 8 1113, Sofia, Bulgaria
e-mail: velin_andonov@math.bas.bg

S. Poryazov
e-mail: stoyan@math.bas.bg

E. Saranova
e-mail: emiliya@math.bas.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_18

(2) credible, (3) feasible and (5) useful. Some important features of the conceptual models are:

- The model is designed for a specific *purpose* and without knowing this purpose it is impossible to create an appropriate simplification.
- *Simplifications* are incorporated in the model to enable more rapid model development and use, and to improve transparency.
- *Assumptions* are made when there are uncertainties or beliefs about the real world being modelled.

The characteristics in [13] are not quantifying.

Eric (see [5]) considers the Universe of Discourse (UoD), describing which classes of entities and propositions are of importance for an application area. UoD consists of: functional and existence dependencies, attributes, subtype-connections, classes, labels. Let S be a concrete conceptual schema (diagram, model) which is to be evaluated by a proposed evaluation function of the following quality measures (see [5]):

- Number of functional dependencies that hold in the UoD, but which are not expressed in S.
- Number of existence dependencies that hold in the UoD, but which are not expressed in S.
- Number of attributes and subtype-connections in S.
- Number of classes in S.
- Number of labels of S.

In [8], the considered metrics for Structural Complexity are:

- Number of Associations (Total number of associations in a model).
- Number of Dependencies (This metrics is used to calculate the total number of dependency relationships within the class diagram).
- Number of Aggregations (It calculates the number of aggregation relationships within a class diagram).
- Depth Inheritance Tree (It calculates the longest path from the class to the root of the hierarchy in a generalization hierarchy).

Metrics for Modularity are [8]:

- Cohesion—this metric calculates the cohesion of different modules.
- Coupling—it calculates the coupling between different modules.

Many of the metrics above are difficult to be evaluated automatically. In the present paper, a more formal approach to metrics of structural complexity is used.

Generalized Nets (GNs, see [4]) are extensions of Petri Nets (see [10]). For many types of Petri Nets and their extensions, it is proven that the functioning and the results of their work can be represented by an ordinary GN [3]. GNs and their extensions allow for detailed modeling of parallel and distributed processes. An important property of the GNs is that one and the same process can be modeled by

more than one GN. As a result, a problem arises of choosing the most suitable GN model of a particular process among the many possible.

In the paper, we study the problem for conceptual optimization of GN models. It is based on operators for complexity of GNs some of which are defined in [3], while some others are defined here for the first time. Relations of inclusion with regard to the results of the work of GNs about the operators are defined which allow for the comparison of the GN models. The optimization is demonstrated for a GN model of a queuing system with finite capacities of the server and buffer, and FIFO (First-In, First-Out) discipline of service of the requests. The choice of the model is justified by the fact that many GN models of queuing systems exist (see [11, 14, 15]). If we want to include such model of a queuing system in a large conceptual model, for instance of an overall telecommunication system (see [12]), it is important to optimize the model, i.e., to represent the system in the simplest possible way without losing conceptual and graphical clarity.

2 On the Concepts in Generalized Nets Models

The GN is a relatively complex object. Detailed definition of a *transition of GN*, *GN* and the algorithms for transition and net functioning can be found in [4]. The concepts of a GN model can be divided into:

- model description concepts;
- graphical representation concepts.

First, we shall describe non-formally the elements used in the graphical representation of a GN. GN's *places* are represented by ◯. A *transition* is a part of the net whose graphical representation looks like the object shown in Fig. 1. Every transition contains *transition's conditions* which are graphically represented by ▼|.

Like Petri nets, GNs contain tokens which are transferred from place to place through the *arcs* of the net. The arcs are denoted by arrows in Fig. 1. They begin at a place and end at the transition's condition or begin at a transition's condition and end at a place.

The names of the transitions and the places are also included in the graphical representation of the GN model. They can be very important for the understanding of the model by non-specialists in the area of GNs and for the users in general.

To summarize, the concepts of a GN model which are represented graphically are: *transition, place, arc* and the names of the transitions and the places.

Definition 1 Transition of a GN is the following ordered seven-tuple:

$$Z = \langle\langle L', L'', t_1, t_2, r, M, \square \ \rangle\rangle \,, \tag{1}$$

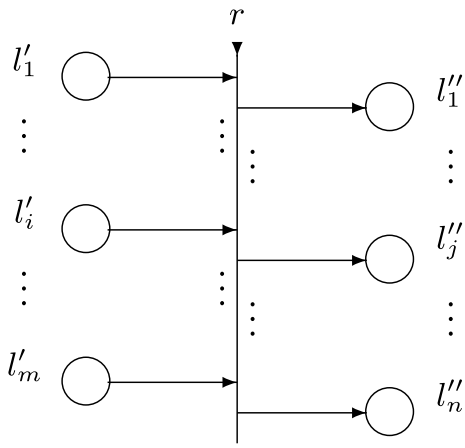


Fig. 1 Graphical representation of a transition of a GN

where

1. L' and L'' are finite non-empty sets of the transition's input and output places, respectively. For the transition in Fig. 1 these sets are:

$$L' = \{l'_1, l'_2, \ldots, l'_m\},$$

$$L'' = \{l''_1, l''_2, \ldots, l''_n\}.$$

2. t_1 is the current time moment at which the transition can be activated.
3. t_2 is the duration of the active state of the transition.
4. r is the IM of the transition's conditions which determine the output places to which the tokens in the input places can be transferred. It has the form:

$$r = \begin{array}{c|c} & l''_1 \ \ldots \ l''_j \ \ldots \ l''_n \\ \hline l'_1 & \\ \vdots & \\ l'_i & \begin{array}{c} r_{i,j} \\ (r_{i,j} \ - \ \text{predicate}) \end{array} \\ \vdots & (1 \le i \le m, 1 \le j \le n) \\ l'_m & \end{array},$$

   where $r_{i,j}$ is the predicate which expresses the condition for transfer from the i-th input place to the j-th output place.
   When $r_{i,j}$ has truth-value "$true$", then a token from the i-th input place can be transferred to the j-th output place. Otherwise, this is impossible.
5. M is an IM of the arcs' capacities. It has the form:

$$M = \begin{array}{c|c} & l''_1 \ \ldots \ l''_j \ \ldots \ l''_n \\ \hline l'_1 & \\ \vdots & \\ l'_i & m_{i,j} \\ & (m_{i,j} \geq 0 - \text{ natural number or } \infty) \\ \vdots & (1 \leq i \leq m, 1 \leq j \leq n) \\ l'_m & \end{array} ;$$

6. $\square$ is the transition type. It is an object having a form similar to a Boolean expression. It has as variables the same symbols that serve as labels for the transition's input places. It is constructed of these variables and the Boolean connectives $\wedge$ and $\vee$. When the value of its type (calculated as a Boolean expression) is "$true$", the transition can become active. Otherwise, it cannot.

Definition 2 *Generalized net E* is the ordered four-tuple

$$E = \langle\langle A, \pi_A, \pi_L, c, f, \theta_1, \theta_2\rangle, \langle K, \pi_K, \theta_K\rangle, \langle T, t^0, t^*\rangle, \langle X, \Phi, b\rangle\rangle , \qquad (2)$$

where

1. A is the set of GN-transitions.
2. π_A is a function which gives the priorities of the transitions, i.e., $\pi_A : A \to \mathcal{N}$.
3. π_L is a function which gives the priorities of the places, i.e., $\pi_L : L \to \mathcal{N}$, where

$$L = pr_1 A \cup pr_2 A$$

   is the set of all GN-places.
4. c is a function which gives the capacities of the places, i.e., $c : L \to \mathcal{N}$.
5. f is a function which calculates the truth values of the predicates of the transition's conditions. It obtains the values "$false$" or "$true$", or values from the set $\{0, 1\}$. if $\mathcal{P}$ is the set of the predicates used in a given GN-model, then f can be defined as $f : \mathcal{P} \to \{0, 1\}$.
6. θ_1 is a function which gives the next time-moment when a given transition Z can be activated, i.e., $\theta_1(t) = t'$, where $pr_3 Z = t, t' \in [T, T + t^*]$ and $t \leq t'$. The value of this function is calculated at the moment when the transition terminates its functioning.
7. θ_2 is a function which gives the duration of the active state of a given transition Z, i.e., $\theta_2(t) = t'$, where $pr_4 Z = t \in [T, T + t^*]$ and $t' \geq 0$. The value of this function is calculated at the moment when the transition starts functioning.
8. K is the set of the GN's tokens. In some cases, it is convenient to consider this set in the form

$$K = \bigcup_{l \in Q^I} K_l,$$

   where K_l is the set of tokens which enter the net from place l, and Q^I is the set of all input places of the net.

9. π_K is a function which gives the priorities of the tokens, i.e., $\pi_K : K \to \mathcal{N}$.
10. θ_K is a function which gives the time moment when a given token can enter the net, i.e., $\theta_K(\alpha) = t$, where $\alpha \in K$ and $t \in [T, T + t^*]$.
11. T is the time-moment when the GN starts functioning; this moment is determined with respect to a fixed (global) time-scale.
12. t^0 is an elementary time-step, related to the fixed (global) time-scale.
13. t^* is the duration of the GN's functioning;
14. X is a function which assigns initial characteristics to each token when it enters an input place of the net.
15. Φ is a characteristic function which assigns new characteristics to each token when it makes a transfer from an input to an output place of a given transition;
16. b is a function which gives the maximum number of characteristics a given token can receive, i.e., $b : K \to N$.

The input and output places of the transitions, the IMs of the capacities of the arcs and the types of the transitions determine the static structure of a GN. The dynamic nature of a GN is expressed in the tokens and the transitions' condition. The temporal properties of a GN are presented through the time components T, t^0, t^* and the elements of the set $pr_{3,4}A$, i.e., the functions θ_1 and θ_2 of the transitions. The functions Φ, X and b serve as a GN's memory. The functions π_A, π_L, c are related to the GN's static structure, f and π_K are related to the GN's dynamic elements and θ_1, θ_2 and θ_K are related to the time components of a GN.

Let

$$\begin{aligned}\Omega = \{&A, \pi_A, \pi_L, c, f, \theta_1, \theta_2, K, \pi_K, \theta_K, T, t^0, t^*, X, \Phi, b\}\\ &\cup\{A_i | 1 \le i \le 7\},\end{aligned} \tag{3}$$

where $A_i = pr_i A$ is the i-th projection of the set of transitions A of a the GN. The elements of the set Ω are the model description concepts of the ordinary GNs.

GNs which have more concepts in their definition form special classes of *extensions of the ordinary GNs*. Many such extensions are described in [4]. The use of more concepts in the extensions makes the modelling of particular real processes easier. All extensions of the ordinary GNs have been proved to be conservative, i.e., the functioning and results of work of every net from the extensions can be represented by an ordinary GN (see [3, 4]).

GNs which have less concepts than the ordinary GNs form classes of *reduced GNs*. If Σ, as usual, is the class of all GNs and $Y \in \Omega$, then by Σ^Y we denote the class of all GNs that do not have component Y. We call them Y-reduced GNs. Many assertions for the various classes of reduced GNs are proved in [3]. The class $\Sigma^* = \Sigma^{A_3, A_4, A_6, A_7, \pi_A, \pi_L, c, \theta_1, \theta_2, \pi_K, \theta_K, T, t^0, t^*, b}$ is the class of minimal reduced GNs (*-GNs). The minimal reduced GNs have the form:

$$E' = \langle A', K^*, X, \Phi \rangle .$$

The minimal elements of Σ^* are denoted by

$$E^* = \langle A^*, K^*, X^*, \Phi^* \rangle ,$$

where A^* is a set of transitions of the form $Z^* = \langle L', L'', r \rangle$.

It is proved that there exist three minimal (i.e., using the least possible number of concepts) reduced classes of GNs (see [1]).

The concepts used in the formal description of the various types of GNs are important characteristics of the net. Therefore, it is useful to introduce the following notation:

$$\Omega_E = \{C | C \in \Omega \ \& \ C \text{ is an element of } E\}. \tag{4}$$

Obviously, when E is an ordinary GN $\Omega_E \equiv \Omega$.

3 Operators for Complexity of GN Models

Many operators are defined over GNs which are divided into six groups: global, local, hierarchical, reducing, extending and dynamical [4]. They transform one GN into another with different components and different graphical representation. Since different GNs can be used to model one and the same process, the notion of complexity of GNs is very important. Depending on the purpose of the model, a more simple or more detailed GN can be chosen.

Some operators for complexity of GNs are defined in [3]. Below, we briefly present some of them and propose new ones. For arbitrary transition Z (see (1)) and arbitrary GN E (see (2)):

- $\phi_1(E) = |pr_1 pr_1 E|$ is the number of the transitions of the net;
- $\phi_2(E) = |pr_1 pr_1 pr_1 E \cup pr_2 pr_1 pr_1 E|$ is the number of places of the net;
- $\phi_3(E) = |pr_1 pr_2 E|$ is the number of tokens of the net;
- $\phi_4(E) = |pr_3 pr_3 E|$ is the duration of the GN functioning;
- $$\phi_5(E) = \sum_{Z \in pr_1 pr_1 E} \phi'_5(Z) = \sum_{k=1}^{|L'|} \sum_{k=l}^{|L''|} k \cdot l \binom{|L'|}{k} \binom{|L''|}{l}$$

  is operator for the complexity of the transitions of the net;
- $\phi_6(E) = \max_{\alpha \in pr_1 pr_2 E} b(\alpha)$ is the maximum number of characteristics that the tokens can keep during the functioning of the net;
- $\phi_7(E) = |pr_1 pr_4 E|$ is the number of initial characteristics of the tokens;
- $\phi_8(E) = \sum_{Z \in pr_1 pr_1 E} pr_1 Z pr_2 Z$ is the number of arcs of the net;
- $\phi_9(E) = |\bigcup_{Z \in pr_1 pr_1 E} \{l | l \in pr_1 Z \ \& \ l \in pr_2 Z\}|$ is the number of places which are both intput and output for a given transition, i.e., the number of loops. This operator gives us information about the graphical representation of the net.

- $\phi_{10}(E) = \dfrac{\sum_{r \in pr_5 pr_1 pr_1 E} |\{r_{i,j} | r_{i,j} \in r \& (r_{i,j} = false \vee r_{i,j} = true)\}|}{\sum_{r \in pr_5 pr_1 pr_1 E} |\{r_{i,j} | r_{i,j} \in r\}|}$ is operator of determinacy, i.e., a ratio of the number of elements of the IMs of the predicates of the transitions with truth values "true" or "false" to the total number of predicates.
- $\phi_{11}(E) = |\Omega_E|$ is the number of concepts used to describe the GN E;
- $\phi_{12}(E)$ is the number of concepts used in the graphical representation of the GN E.

Different GNs can be compared with regard to these operators. In the process of comparison of GNs and selection of a better one for the particular modeling purpose, it is important that relations regarding the functiong and results of the work of the nets exist. Among the many relations defined over GNs, the most important to the results of the work of the nets are the realtions of *inclusion with respect to the final characteristics of the tokens* (denoted by $\sqsubset$) and *inclusion with respect to all characteristics of the tokens* (denoted by $\sqsubset_*$). To define them we need the following notation:

$$E(\alpha, x) = \begin{cases} \langle \alpha, x_{\alpha_{fin}} \rangle & , \text{ if } \alpha \in K(E) \,\&\, x \in X(\alpha). \\ \langle \alpha, x \rangle & , \text{ else.} \end{cases}$$

$$E\{\alpha, x\} = \begin{cases} \langle \alpha, x, x_{\alpha_1}, \ldots, x_{\alpha_{fin}} \rangle & , \text{ if } \alpha \in K(E) \,\&\, x \in X(\alpha)\,. \\ \langle \alpha, x \rangle & , \text{ else}\,. \end{cases}$$

For two GNs E_1 and E_2 the following relations are defined in [3].

Definition 3 $E_1 \sqsubset E_2 \;\; \underline{iff} \;\; (K_1(E_1) \subset K_2(E_2) \,\&\, (X_1(E_1) \subset X_2(E_2))) \,\&$ $(\forall \alpha \in K_1(E_1))(\forall x \in X_1(E_1))(E_1(\alpha, x) = E_1(\alpha, x))\,.$

Definition 4 $E_1 \sqsubset_* E_2 \;\; \underline{iff} \;\; (K_1(E_1) \subset K_2(E_2) \,\&\, (X_1(E_1) \subset X_2(E_2))) \,\&$ $(\forall \alpha \in K_1(E_1))(\forall x \in X_1(E_1))(E_1\{\alpha, x\} = E_2\{\alpha, x\})\,.$

Definition 5 $E_1 \approx E_2 \;\; \underline{iff} \;\; E_1 \sqsubset E_2 \,\&\, E_2 \sqsubset E_1\,.$

Definition 6 $E_1 \approx_* E_2 \;\; \underline{iff} \;\; E_1 \sqsubset_* E_2 \,\&\, E_2 \sqsubset_* E_1\,.$

If ϕ is some operator for complexity, then we can define relations of inclusion and equivalence between GNs with respect to the operator in the following way:

Definition 7 $E_1 \vdash_\phi E_2 \equiv (E_1 \approx E_2 \,\&\, \phi(E_1) \geq \phi(E_2)) \vee (E_1 \sqsubset E_2)\,.$

Definition 8 $E_1 \approx_\phi E_2 \equiv (E_1 \approx E_2) \,\&\, (\phi(E_1) \geq \phi(E_2))\,.$

Similar relations can be defined about the relations $\sqsubset_*$ and $\approx_*$.

4 Conceptual Optimization of a GN Model of a Queuing System

Since different GNs can be used to model one and the same process, it is important to determine which one is the best with regard to the purpose of the modelling. As it is shown in [3], a given GN can be modified through the operators over GNs. As a result of the application of some of the operators, the resulting net can have less (or more) transitions, places, tokens, etc. Suppose we have a GN with high value of some operator for complexity ϕ_i which we want to simplify. By applying the operators to it, we can obtain a sequence of GNs $E, E_1, \ldots, E_n$ such that

$$E \approx_{\phi_i} E_1 \approx_{\phi_i} \ldots \approx_{\phi_i} E_n. \tag{5}$$

This process can continue until we obtain a GN with one transition and two places, which would be minimal with respect to the operator. However, such GN is not very useful. Therefore, this process must be terminated at some point when the last obtained GN is the most optimal one. Specifying the optimization criteria, for example the threshold value of the corresponding operator for complexity, is a problem from the methodological aspect of the theory of the GNs. The modeller should determine the optimal number of transitions, places, tokens, etc., of the GN which give the optimal values of the complexity operator (or collection of operators).

4.1 First GN Model of a Queuing System

To illustrate the optimization of a GN, we consider a queuing system [6] consisting of buffer and server with finite capacities and FIFO (First-In, First-Out) discipline of service of the requests by the server. GN models of queuing systems with various disciplines of service of the requests are described in [14, 15]. A comparison between the GN approach to the conceptual modelling of queuing systems and the Service Systems Theory approach is made in [11]. Below we use some of the results presented in these papers.

First, we propose a detailed GN model of a queuing system with graphical representation shown in Fig. 2. It corresponds to a detailed conceptual model of queuing system proposed in [11], which uses elements of Service Systems Theory.

The GN consists of 7 transitions and $14 + Nq$ places where Nq is the capacity of the buffer. The transitions represent the following functions of the queuing system:

- Z_1 represents the process of generating of requests.
- Z_2 represents the blocking of the requests when the buffer has reached its capacity.
- Z_3 determines the way of service of the requests, i.e., with waiting or without waiting.
- Z_4 represents the service of the requests without delay, when the server has not reached its capacity.

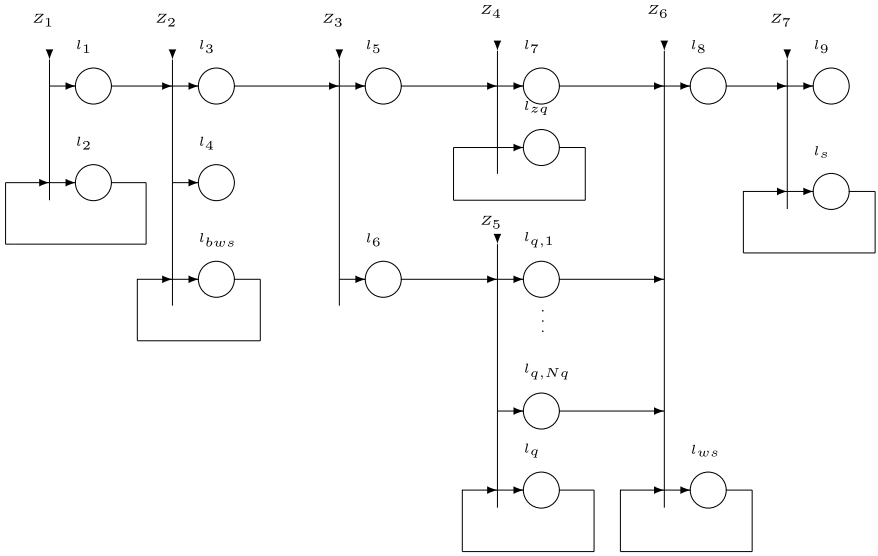


Fig. 2 First GN model of a queuing system

- Z_5 represents the service of the requests with waiting, when the server has reached its capacity.
- Z_6 represents the function of the buffer of the queuing system.
- Z_7 represents the function of the server of the queuing system.

A special naming system of the important places in which tokens of the GN collect the values of the parameters of the queuing system is used. Their meaning is the following:

- l_{bws}: "bws" is an abbreviation of "blocked waiting for switch".
- l_{zq}: "zq" is an abbreviation of "zero queuing"; represents the service of the requests without delay.
- $l_{q,1}, \ldots, l_{q,Nq}, l_q$: waiting places of the buffer, where Nq is the buffer capacity.
- l_{ws}: "ws" is an abbreviation of "waiting for switch"; represents the buffer of the queuing system.
- l_s: "s" is an abbreviation of "server"; represents the server of the queuing system.

For every such place l_x, the following parameters are used in the model see [12]:

- Fx—intensity or incoming rate (frequency) of the flow of requests (i.e. the number of requests per time unit);
- Px—probability of directing the requests towards l_x;
- T_x—service time (duration of servicing of a request);
- Yx—traffic intensity [Erlang];
- Vx—traffic volume [Erlang - time unit];
- Nx—number of lines (service resources, positions, capacity).

Six types of tokens are used in the model:

- Tokens of type α represent the requests. In the initial time moment of the GN functioning, token α stays in place l_2 with initial characteristic "formula for generating the flow of requests".
- Token of type β stays in place l_{bws} in the initial time moment with initial characteristic "initial values of Y_{bws}, P_{bws}, F_{bws}, T_{bws}, N_{bws}".
- Token of type γ stays in place l_{zq} in the initial time moment with initial characteristic "initial values of Y_{zq}, P_{zq}, F_{zq}, T_{zq}, N_{zq}".
- Token of type δ stays in place l_q in the initial time moment with initial characteristic "initial values of Y_q, P_q, F_q, T_q, N_q". The discipline of service of the requests can be specified in this initial characteristic. We consider only FIFO discipline of service of the requests.
- Token of type ϵ stays in place l_{ws} in the initial time moment with initial characteristic "initial values of Y_{ws}, P_{ws}, F_{ws}, T_{ws}, N_{ws}".
- Token of type ζ stays in place l_s in the initial time moment with initial characteristic "initial values of Y_s, P_s, F_s, T_s, N_s".

We shall give the formal description of the transitions of the GN. The GN is a reduced one (see [4]). The transitions of the reduced GN have the form $Z = \langle L', L'', r\rangle$ where L' is the set of the input places of the transition, L'' is the set of output places of the transition and r is the index matrix of the transition's condition.

$$Z_1 = \langle \{l_2\}, \{l_1, l_2\}, r_1\rangle,$$

where

$$r_1 = \begin{array}{c|cc} & l_1 & l_2 \\ \hline l_2 & W_{2,1} & W_{2,2} \end{array}$$

and

- $W_{2,1}$ ="A flow of requests is generated";
- $W_{2,2}$ ="true".

When the truth value of the predicate $W_{2,1}$ is "true" the α token in place l_2 splits into two tokens—the same α token which stays in place l_2 and token α_1 which enters place l_1 with characteristic "*volume, duration of service, calls destinations*". The α token in place l_2 obtains the characteristic "*current formula for generation of the requests*".

$$Z_2 = \langle \{l_1, l_{bws}\}, \{l_3, l_4, l_{bws}\}, r_2\rangle,$$

where

$$r_2 = \begin{array}{c|ccc} & l_3 & l_4 & l_{bws} \\ \hline l_1 & W_{1,3} & W_{1,4} & W_{1,bws} \\ l_{bws} & W_{bws,3} & W_{bws,4} & W_{bws,bws} \end{array}$$

and

- $W_{1,3} =$ "$Y_{ws} < N_{ws}$";
- $W_{1,4} = \neg W_{1,3} = W_{1,bws}$;
- $W_{bws,bws}$ = "true";
- $W_{bws,3} = W_{bws,4}$ = "false".

When the truth value of the predicate $W_{1,3}$ is "true", the α_1 token from place l_1 enters place l_3 without obtaining new characteristic. When the truth value of the predicate $W_{1,4}$ is "true" the token of type α in place l_1 splits into two identical tokens one of which enters place l_4 and the other one merges with the β token in place l_{bws}. In place l_4 the tokens do not obtain new characteristics. Token β in place l_{bws} obtains the characteristic "*current value of* $Ybws$".

$$Z_3 = \langle \{l_3\}, \{l_5, l_6\}, r_3 \rangle,$$

where

$$r_3 = \begin{array}{c|cc} & l_5 & l_6 \\ \hline l_3 & W_{3,5} & W_{3,6} \end{array}$$

and

- $W_{3,5} =$"$Y_s < N_s$";
- $W_{3,6} = \neg W_{3,5}$.

When the truth value of the predicate $W_{3,5}$ is "true", the α token from place l_3 enters place l_5 without obtaining new characteristic. When the truth value of the predicate $W_{3,6}$ is "true" the token of type α in place l_3 enters place l_5 without obtaining new characteristic.

$$Z_4 = \langle \{l_5, l_{zq}\}, \{l_7, l_{zq}\}, r_4 \rangle,$$

where

$$r_4 = \begin{array}{c|cc} & l_7 & l_{zq} \\ \hline l_5 & W_{5,7} & W_{5,zq} \\ l_{zq} & W_{zq,7} & W_{zq,zq} \end{array}$$

and

- $W_{5,7} = W_{5,zq} = W_{zq,zq} =$"true";
- $W_{zq,7} =$"false".

The token of type α in place l_5 splits into two identical tokens one of which enters place l_7 without obtaining new characteristic. The other one merges with the γ token in place l_{zq}. The γ token in place l_{zq} obtains the characteristic *"current values of* $Y_{zq}, P_{zq}, F_{zq}, T_{zq}, N_{zq}$".

$$Z_5 = \langle \{l_6, l_q\}, \{l_{q,1}, \ldots, l_{q,Nq}, l_q\}, r_5 \rangle,$$

where

$$r_5 = \begin{array}{c|cccc} & l_{q,1} & \cdots & l_{q,Nq} & l_q \\ \hline l_6 & W_{6,q1} & \cdots & W_{6,qNq} & W_{6,q} \\ l_q & W_{q,q1} & \cdots & W_{q,qNq} & W_{q,q} \end{array}$$

and

- $W_{6,qi}$ =“The output place $l_{q,i}$ is the highest priority empty place among the places $l_{q,1}, l_{q,2}, \ldots, l_{q,Nq}$”, for $i = 1, 2, \ldots, Nq$;
- $W_{6,q} = W_{6,q1} \vee W_{6,q2} \vee \ldots \vee W_{6,qNq}$;
- $W_{q,qi}$ = “false”, for $i = 1, 2, \ldots, Nq$;
- $W_{q,q}$ =“true”.

When the truth value of one of the predicates $W_{6,qi}$ for $i = 1, 2, \ldots, Nq$ is “true” the token of type α in place l_6 splits into two identical tokens one of which enters the corresponding output place without obtaining new characteristic while the other merges with the δ token in place l_q. The δ token in place l_q obtains the characteristic “*current value of* Y_q, T_q, F_q; *list of all tokens in the output places* $l_{q,1}, l_{q,2}, \ldots, l_{q,Nq}$ *and the duration of their stay in the corresponding place*”.

$$Z_6 = \langle \{l_7, l_{q,1}, \ldots, l_{q,Nq}, l_{ws}\}, \{l_8, l_{ws}\}, r_6 \rangle,$$

where

$$r_6 = \begin{array}{c|cc} & l_8 & l_{ws} \\ \hline l_7 & W_{7,8} & W_{7,ws} \\ l_{q,1} & W_{q1,8} & W_{q1,ws} \\ \vdots & \vdots & \vdots \\ l_{q,Nq} & W_{qNq,8} & W_{qNq,ws} \\ l_{ws} & W_{ws,8} & W_{ws,ws} \end{array}$$

and

- $W_{7,8} = W_{7,ws} = W_{ws,ws}$ =“true”;
- $W_{qi,8}$ =“the current token has stayed more time in the place than the tokens in all other places $l_{q,1}, l_{q,2}, \ldots, l_{q,Nq}$ and $Y_s < N_s$”, for $i = 1, 2, \ldots, Nq$;
- $W_{qi,ws} = W_{qi,8}$, for $i = 1, 2, \ldots, Nq$;
- $W_{ws,8}$ = “false”.

The α token in place l_7 splits into two tokens one of which enters place l_8 without obtaining new characteristic and the other one enters l_{ws}. When the truth value of one of the predicates $W_{qi,8}$ is “true” the token of type α from the corresponding place $l_{q,i}$ splits into two identical tokens one of which enters place l_8 without obtaining new characteristic and another one which enters place l_{ws}. The tokens entering place l_{ws} merge with the δ token in place l_{ws}. The δ token obtains the characteristic “*current value of* Yws, Tws, Fws”.

$$Z_7 = \langle \{l_8, l_s\}, \{l_9, l_s\}, r_7 \rangle,$$

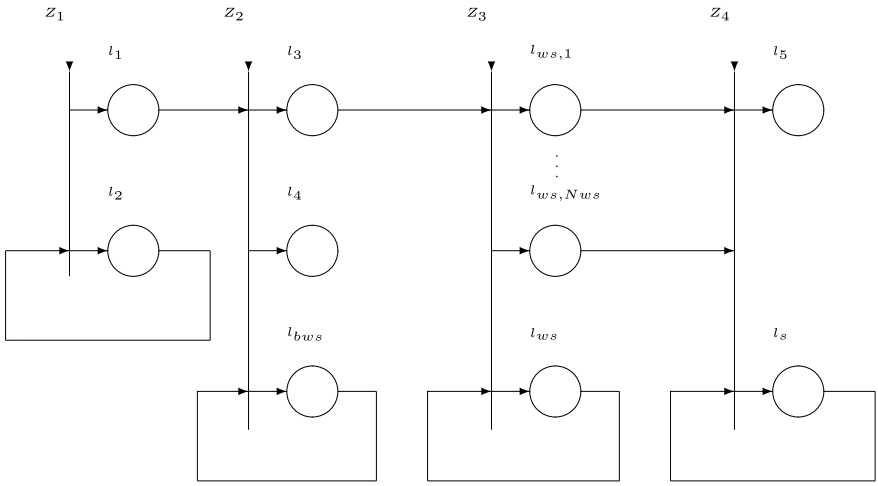


Fig. 3 Second GN model of a queuing system

where

$$r_7 = \begin{array}{c|cc} & l_9 & l_s \\ \hline l_8 & W_{8,9} & W_{8,s} \\ l_s & W_{s,9} & W_{s,s} \end{array}$$

and

- $W_{8,9}$ =“false”;
- $W_{8,s} = W_{s,9} = W_{s,s}$ = “true”.

The token of type α from place l_8 enters place l_s where it merges with the ζ token. The ζ token in place l_s splits into two identical tokens one of which enters place l_9 with characteristic “*flow intensity of serviced requests by the queuing system*”. The ζ token in place l_s obtains the characteristic “*current value of* Ys, Ts, Fs”.

Let E_1 be the GN described above. Then $\phi_1(E_1) = 7$, $\phi_2(E_1) = 14 + Nq$, $\phi_3(E_1) = 6$, $\phi_7(E_1) = 6$, $\phi_8(E_1) = 24 + 4Nq$, $\phi_9(E_1) = 6$,

$$\phi_{10}(E_1) = \frac{17 + Nq}{24 + 4Nq},$$

$\phi_{11}(E) = 6$, $\phi_{12}(E) = 5$.

4.2 Second GN Model of a Queuing System

The GN E_1 represents the most detailed representation of a queuing system with FIFO discipline of service of the requests. From the graphical representation of the

GN are also evident the two ways of service of the requests: with waiting and without waiting. Here, we modify this model, by substituting the three transitions Z_3, Z_4 and Z_5 with a single transition which represents the function of the buffer. The new GN is shown in Fig. 3.

Transitions Z_1 and Z_2 are the same as in the first GN model. Transitions Z_3 and Z_4 are different compared to those in the first GN model. Transition Z_3 represents the function of the buffer. Transition Z_4 represents the function of the server. The waiting places of the buffer are represented by places $l_{ws,1}, l_{ws,2}, \ldots, l_{ws,Nws}$, where Nws is the buffer capacity and place l_{ws} is used to store the values of the parameters of the buffer device. Four types of tokens are used:

- Tokens of type α represent the requests. In the initial time moment of the GN functioning, token α stays in place l_2 with initial characteristic "formula for generating the flow of requests".
- Token of type β stays in place l_{bws} in the initial time moment with initial characteristic "initial values of Y_{bws}, P_{bws}, F_{bws}, T_{bws}, N_{bws}".
- Token of type γ stays in place l_{ws} in the initial time moment with initial characteristic "initial values of Y_{ws}, P_{ws}, F_{ws}, T_{ws}, N_{ws}".
- Token of type δ stays in place l_s in the initial time moment with initial characteristic "initial values of Y_s, P_s, F_s, T_s, N_s".

We shall give only the formal description of transitions Z_3 and Z_4. The GN is again a reduced one (see [4]).

$$Z_3 = \langle \{l_3, l_{ws}\}, \{l_{ws,1}, \ldots, l_{ws,Nws}, l_{ws}\}, r_3 \rangle,$$

where

$$r_3 = \begin{array}{c|cccc} & l_{ws,1} & \cdots & l_{ws,Nws} & l_{ws} \\ \hline l_3 & W_{3,ws1} & \cdots & W_{3,wsNws} & W_{3,ws} \\ l_{ws} & W_{ws,ws1} & \cdots & W_{ws,wsNws} & W_{ws,ws} \end{array}$$

and

- $W_{3,wsi}$ ="The output place $l_{ws,i}$ is the highest priority empty place among the places $l_{ws,1}, l_{ws,2}, \ldots, l_{ws,Nws}$", for $i = 1, 2, \ldots, Nws$;
- $W_{3,ws} = W_{3,ws1} \vee W_{3,ws2} \vee \ldots \vee W_{3,wsNws}$;
- $W_{ws,wsi}$ = "false", for $i = 1, 2, \ldots, Nws$;
- $W_{ws,ws}$ = "true".

When the truth value of one of the predicates $W_{3,wsi}$ for $i = 1, 2, \ldots, Nws$ is "true" the token of type α in place l_3 splits into two identical tokens one of which enters the corresponding output place without obtaining new characteristic while the other merges with the γ token in place l_{ws}. The γ token in place l_{ws} obtains the characteristic "*current value of* Y_{ws}, T_{ws}, F_{ws}; *list of all tokens in the output places* $l_{ws,1}, l_{ws,2}, \ldots, l_{ws,Nws}$ *and the duration of their stay in the corresponding place; values of* Y_{zw}, T_{zw}, F_{zw} *for the requests serviced without waiting*".

$$Z_4 = \langle \{l_{ws,1}, \ldots, l_{ws,Nws}, l_s\}, \{l_5, l_s\}, r_4 \rangle,$$

where

$$r_4 = \begin{array}{c|cc} & l_5 & l_s \\ \hline l_{ws,1} & W_{ws1,5} & W_{ws1,s} \\ \vdots & \vdots & \vdots \\ l_{ws,Nws} & W_{wsNws,5} & W_{wsNws,s} \\ l_s & W_{s,5} & W_{s,s} \end{array}$$

and

- $W_{wsi,5}$ = "false", for $i = 1, 2, \ldots, Nws$;
- $W_{wsi,s}$ = "the current token has stayed more time in the place than the tokens in all other places $l_{ws,1}, l_{ws,2}, \ldots, l_{ws,Nws}$ and $Y_s < N_s$", for $i = 1, 2, \ldots, Nws$;
- $W_{s,5} = W_{s,s}$ = "true".

When the truth value of one of the predicates $W_{wsi,s}$ is "true" the token of type α from the corresponding place $l_{ws,i}$ enters place l_s where it merges with the δ token. The δ token in place l_s splits into two tokens the same δ token and one of which enters place l_5 with characteristic "*flow intensity of the serviced requests by the queuing system*". The δ token in place l_s obtains the characteristic "*current values of* Ys, Ts, Fs".

Let E_2 be the GN described above. Then we have: $\phi_1(E_2) = 4$, $\phi_2(E_2) = 8 + Nws$, $\phi_3(E_2) = 4$, $\phi_7(E_2) = 4$, $\phi_8(E_2) = 12 + 4Nws$, $\phi_9(E_2) = 4$,

$$\phi_{10}(E_2) = \frac{4 + Nws}{5 + 2Nws},$$

$\phi_{11}(E_2) = 6, \phi_{12}(E_2) = 5$. Since $Nq = Nws$ because the buffer capacity is the same in both GNs, we obtain $E_1 \sqsubset_{\phi_i} E_2$ for $i = 1, 2, 3, 6, 7, 8, 9, 11, 12$.

4.3 Third GN Model of a Queuing System

In the previous two GN models, the transitions representing the function of the buffer have one place for every waiting place of the buffer. For queuing systems with low buffer capacity this is a convenient representation, especially for the graphical representation. For queuing systems with large buffer capacities (or infinite) that is not optimal representation. It is possible to substitute the waiting places with one place, as in the third GN model shown in Fig. 4.

Each of the transitions of the GN has the same function as in the second GN. The tokens and their initial characteristics are also the same. We shall describe transitions Z_3 and Z_4 which are different from their corresponding transitions in the second GN.

$$Z_3 = \langle \{l_3, l_{ws}\}, \{l_5, l_{ws}\}, r_3 \rangle,$$

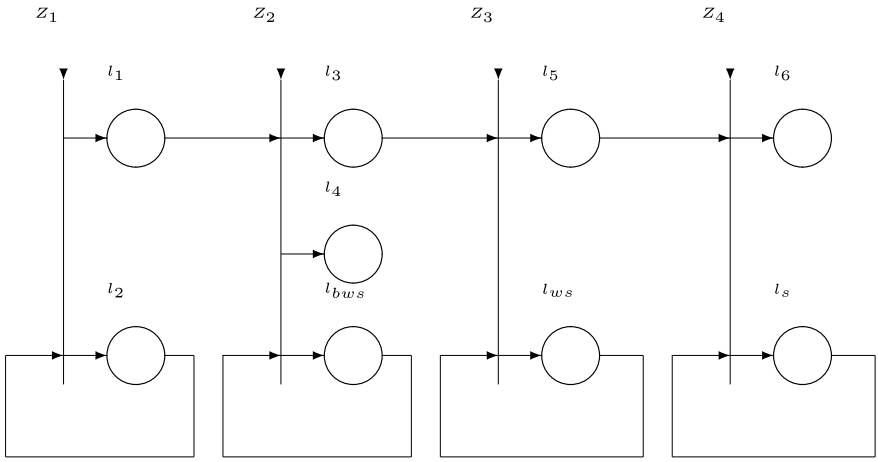


Fig. 4 Third GN model of a queuing system

where

$$r_3 = \begin{array}{c|cc} & l_5 & l_{ws} \\ \hline l_3 & W_{3,5} & W_{3,ws} \\ l_{ws} & W_{ws,5} & W_{ws,ws} \end{array}$$

and

- $W_{ws,5}$ =“false”;
- $W_{3,ws} = W_{3,5} = W_{ws,ws}$ = “true”.

The token of type α from place l_3 splits into two tokens one of which enters place l_5 without obtaining new characteristic. The other one enters place l_{ws} where it merges with the γ token. The γ token in place l_{ws} obtains the characteristic “*current value of* Y_{ws}, T_{ws}, F_{ws}; *list of all tokens in the output place* l_5 *and the duration of their stay in the place; values of* Y_{zw}, T_{zw}, F_{zw}”.

$$Z_4 = \langle \{l_5, l_s\}, \{l_6, l_s\}, r_4 \rangle,$$

where

$$r_4 = \begin{array}{c|cc} & l_6 & l_s \\ \hline l_5 & W_{5,6} & W_{5,s} \\ l_s & W_{s,6} & W_{s,s} \end{array}$$

and

- $W_{s,6}$ =“false”;
- $W_{5,6} = W_{5,s}$ =“the current token has stayed more time in the place than the rest of the tokens and $Y_s < N_s$”;
- $W_{s,s}$ = “true”.

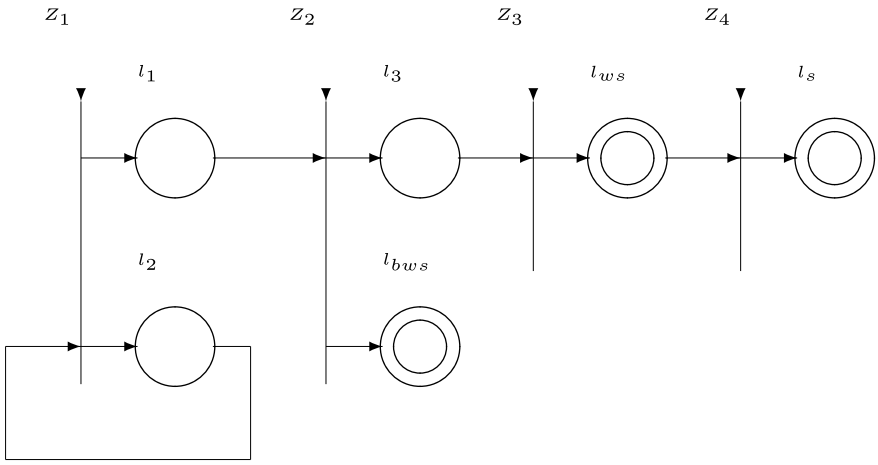


Fig. 5 Fourth GN model of a queuing system

When the truth value of one of the predicates $W_{5,6}$ is "true" the token of type α from place l_5 splits into two identical tokens one of which enters place l_6 with characteristic "*flow intensity of the serviced requests by the queuing system*". The other one enters place l_s where it merges with the δ token. The δ token in place l_s obtains the characteristic "*current values of* Ys, Ts, Fs".

Let E_3 be the GN described above. The operators for complexity have the following values: $\phi_1(E_3) = 4$, $\phi_2(E_3) = 9$, $\phi_3(E_3) = 4$, $\phi_7(E_3) = 4$, $\phi_8(E_3) = 16$, $\phi_9(E_3) = 4$, $\phi_{10}(E_3) = 5/8$, $\phi_{11}(E_3) = 6$, $\phi_{12}(E_3) = 5$. Therefore, we have $E_2 \sqsubset_{\phi_i} E_3$ for $i = 1, 2, 3, 7, 8, 9, 11, 12$.

4.4 Fourth GN Model of a Queuing System

The third GN model can be further optimized if we use the extension of the ordinary GNs—Generalized Nets with Characteristics of the Places (GNCP, [2]). The places which can obtain characteristics are presented with two concentric circles in the graphical representation of the net in Fig. 5. Now, only one type of tokens is used. Tokens of type α represent the requests. In the initial time moment of the GN functioning, token α stays in place l_2 with initial characteristic "formula for generating the flow of requests". The initial characteristics of the places are:

- Place l_{bws} has initial characteristic *"initial values of* Y_{bws}, P_{bws}, F_{bws}, T_{bws}, N_{bws}".
- Place l_{ws} has initial characteristic *"initial values of* Y_{ws}, P_{ws}, F_{ws}, T_{ws}, N_{ws}".
- Place l_s has initial characteristic *"initial values of* Y_s, P_s, F_s, T_s, N_s".

Transition Z_1 is the same as in the previous models. The other transitions are described below.

$$Z_2 = \langle \{l_1\}, \{l_3, l_{bws}\}, r_2 \rangle,$$

where

$$r_2 = \begin{array}{c|cc} & l_3 & l_{bws} \\ \hline l_1 & W_{1,3} & W_{1,bws} \end{array}$$

and

- $W_{1,3}$ =“$Y_{ws} < N_{ws}$”;
- $W_{1,bws}$ =“$\neg W_{1,3}$”.

When the truth value of the predicate $W_{1,3}$ is “true” the α token from place l_1 enters place l_3 without obtaining new characteristic. When the truth value of the predicate $W_{1,ws}$ is “true” the α token from place l_1 enter place l_{bws} without obtaining new characteristic. Place l_{bws} obtains the characteristic “*current value of* $Ybws$”.

$$Z_3 = \langle \{l_3\}, \{l_{ws}\}, r_3 \rangle,$$

where

$$r_3 = \begin{array}{c|c} & l_{bws} \\ \hline l_3 & W_{3,ws} \end{array}$$

and

- $W_{3,ws}$ =“true”.

Token α at place l_3 enters place l_{ws} without obtaining new characteristic. Place l_{ws} obtains the characteristic: “*current value of* Y_{ws}, T_{ws}, F_{ws}; *list of all requests and the duration of their stay in buffer; values of* Y_{zw}, T_{zw}, F_{zw} *for the requests serviced without waiting*”.

$$Z_4 = \langle \{l_{ws}\}, \{l_s\}, r_4 \rangle,$$

where

$$r_4 = \begin{array}{c|c} & l_s \\ \hline l_{ws} & W_{ws,s} \end{array}$$

and

- $W_{ws,s}$ =“true”.

Token α at place l_{ws} enters place l_s without obtaining new characteristic. Place l_s obtains the characteristic: “*current value of* Y_s, T_s, F_s; *list of all requests serviced with FIFO discipline of service*”.

Let E_4 be the GN described above. Then, we have: $\phi_1(E_4) = 4$, $\phi_2(E_4) = 6$, $\phi_3(E_4) = 1$, $\phi_7(E_4) = 1$, $\phi_8(E_4) = 6$, $\phi_9(E_4) = 1$, $\phi_{10}(E_4) = 1/2$, $\phi_{11}(E_4) = 6$, $\phi_{12}(E_4) = 6$. In this case, the relations $E_3 \sqsubset_{\phi_i} E_4$ for $i = 1, 2, 3, 7, 8, 9, 10, 11$.

The last GN is the optimal representation of a queuing system, in the sense that it has the least acceptable number of transitions and places. It is possible to further reduce the number of transitions and places but some of the concepts of the queuing system will not be presented. The text conceptual description of a queuing system has 4 concepts which must be presented in the graphical representation of the GN model. These are: generator, blocked waiting requests bransch, buffer and server. All of them are presented in the fourth GN.

5 Conclusion

The conceptual model optimization needs appropriate indicators of quality. They have to be objective and evaluated predominantly automatically, if we will design optimization algorithms performed by computer. Most of the existing indicators are subjective. In [7], all of the 10 proposed conceptual modeling evaluation criteria are subjective. In this paper, the proposed indicators are suitable for computer evaluation.

The operators for complexity and the relations defined over GNs with respect to these operators are a base for conceptual optimization of GN models. The operators for complexity and the relations over GNs should be generalized to allow comparison of conceptual models based on the GNs theory with conceptual models based on Service Systems Theory.

The comparison of conceptual models presented in different languages, e.g., comparison of conceptual models based on the GNs theory with conceptual models in Service Systems Theory is an extremely challenging task.

Acknowledgements The work of Velin Andonov is supported by the 2020 National Scientific Program for post doctoral and young scientists of the Ministry of Education and Science of Bulgaria.

The work of Stoyan Poryazov is partially supported by the joint research project "Symbolic-Numerical Decision Methods for Algebraic Systems of Equations in Perspective Telecommunication Tasks" of IMI-BAS, Bulgaria and JINR, Dubna, Russia.

The work of Emiliya Saranova was supported by the Task 1.2.5. "Prediction and Guaranteeing of the Quality of Service in Human-Cyber-Physical Systems" of National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)" financed by the Bulgarian Ministry of Education and Science.

References

1. Andonov, V.: Reduced generalized nets with characteristics of the arcs. In: Issues in Intuitionistic Fuzzy Sets and Generalized Nets, vol. 14, pp. 25–35 (2018/19)
2. Andonov, V., Atanassov, K.: Generalized nets with characteristics of the places. Compt. Rend. Acad. bulg. Sci. **66**(12), 1673–1680 (2013)

3. Atanassov, K.: Generalized Nets. World Scientific, Singapore, London (1991)
4. Atanassov, K.: On Generalized Nets Theory. Prof. M. Drinov Academic Publ. House, Sofia (2007)
5. Eick, C.F.: A Methodology for the design and transformation of conceptual schemas. In: Proceedings of the 17th International Conference on Very Large Databases, Barcelona, pp. 25–34 (1991)
6. Giambene, G.: Queuing Theory and Telecommunications, 2nd edn. Springer, US (2014)
7. Loper, M.L., Birta, L.G., Arbez, G.: Lessons from a conceptual modeling exercise. In: Laroque, C., Himmelspach, J., Pasupathy, R., Rose, O., Uhrmacher, A.M. (eds.) Proceedings of the 2012 Winter Simulation Conference. IEEE. 978-1-4673-4780-8/12/2012
8. Mehmood, K., Cherfi, S., Comyn-Wattiau, I.: Data quality through conceptual model quality—reconciling researchers and practitioners through a customizable quality model. Published in ICIQ (2009). http://mitiq.mit.edu/ICIQ/Documents/IQ%20Conference%202009/Papers/2-C.pdf
9. Oren, O.: A method for optimization of a conceptual model. In: 1984 IEEE First International Conference on Data Engineering, Los Angeles, CA, USA, pp. 126–132 (1984)
10. Petri, C.-A.: Kommunication mit Automaten. Ph.D. Thesis, Univ. of Bonn, 1962.; Schriften des Inst. fur Instrument. Math., No. 2, Bonn (1962)
11. Poryazov, S., Andonov, V., Saranova, E.: Comparison of four conceptual models of a queuing system in service networks. In: Proceedings of the 26th National conference with international participation TELECOM 2018. Sofia, 25–26 Oct 2018, pp. 71–77 (2018)
12. Poryazov, S., Saranova, E.: Models of telecommunication networks with virtual channel switching and applications. Academic Publishing House "Prof. M. Drinov", Sofia (2012)
13. Robinson, S.: Conceptual Modelling: Who Needs It? SCS M&S Magazine (2010/n2)
14. Tomov, Z., Krawczak, M., Andonov, V., Dimitrov, E., Atanassov, K.: Generalized net models of queueing disciplines in finite buffer queueing systems. In: Proceedings of the 16th International Workshop on Generalized Nets, Sofia, 10 Feb 2018, pp. 1–9 (2018)
15. Tomov, Z., Krawczak, M., Andonov, V., Atanassov, K., Simeonov, S.: Generalized net models of queueing disciplines in finite buffer queueing systems with intuitionistic fuzzy evaluations of the tasks. Notes Intuit. Fuzzy Sets **25**(2), 115–122 (2019)
16. Vishnevskiy, V.M.: Theoretical foundations of computer networks planning. Tehnosfera, Moscow (2003). (in Russian)

Sensitivity Study of a Large-Scale Air Pollution Model by Using Optimized Latin Hyprecube Sampling

Venelin Todorov, Ivan Dimov, Tzvetan Ostromsky, Zahari Zlatev, Rayna Georgieva, and Stoyan Poryazov

Abstract In this work a systematic procedure for multidimensional sensitivity analysis in the area of air pollution modeling by an optimized latin hypercube sampling has been done. The Unified Danish Eulerian Model (UNI-DEM) is used in our investigation, because this is one of the most advanced large-scale mathematical models that describes adequately all physical and chemical processes. We study the sensitivity of concentration variations of some of the most dangerous air pollutants with respect to the anthropogenic emissions levels and with respect to some chemical reactions rates.

Keywords Sensitivity study · Air pollution model · Latin hypercube

V. Todorov · S. Poryazov
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
e-mail: stoyan@math.bas.bg

V. Todorov (✉) · I. Dimov · T. Ostromsky · R. Georgieva
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

I. Dimov
e-mail: ivdimov@bas.bg

T. Ostromsky
e-mail: ceco@parallel.bas.bg

R. Georgieva
e-mail: rayna@parallel.bas.bg

Z. Zlatev
National Centre for Environment and Energy, University of Århus, Frederiksborgvej 399, P.O. Box 358, 4000 Roskilde, Denmark
e-mail: zz@dmu.dk

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_19

1 Introduction

High levels of pollution can disrupt ecosystems and cause harm to plants, animals and humans. Therefore, it is extremely important to investigate accurately the levels of contamination. It is necessary to know whether the pollution levels are below some critical values and if so—to develop a reliable control system to keep them within these limits. These problems can be successfully addressed by developing and implementing a reliable SA for studying various pollution-related processes. In many geographical areas of our planet, a regional policy on air pollution is being implemented, to control and reduce the levels of pollution crossing national borders, as well as to limit their impact on human health and ecological systems. Environmental protection is among the top priorities worldwide [13, 40]. There are many challenges in this area of research, as it is a painful topic for society as well as an important issue for the health care system.

Mathematical models are used to study and predict the behavior of a variety of complex systems—engineering, physical, economic, social, environmental [14, 34, 41]. They determine the most important quantities that control the state and behavior of a system, as well as the quantitative regularities, that is, the mathematical laws that underlie the change of these quantities. On the other hand, it is of significant importance to develop techniques to determine the reliability, robustness and efficiency of a mathematical model [8, 26, 39]. One such approach is sensitivity analysis (SA). Sensitivity studies are nowadays applied to some of the most complicated mathematical models from various intensively developing areas of application [2, 3, 15–17, 19].

Here we discuss a systematic approach for sensitivity analysis studies in the area of air pollution modelling. The Unified Danish Eulerian Model (UNI-DEM) [42, 43] is used in this particular study. Different parts of the large amount of output data, produced by the model, were used in various practical applications, where the reliability of this data should be properly estimated. Another reason to choose this model as a case study here is its sophisticated chemical scheme, where all relevant chemical processes in the atmosphere are accurately represented. We study the sensitivity of concentration variations of some of the most dangerous air pollutants with respect to the anthropogenic emissions levels and with respect to some chemical reactions rates.

Several sensitivity analysis techniques have been developed and used throughout the years [33]. In general, these methods rely heavily on special assumptions connected to the behaviour of the model (such as linearity, monotonicity and additivity of the relationship between input and output parameters of the model). Among the quantitative methods, variance-based methods are the most often used [32]. The main idea of these methods is to evaluate how the variance of an input or a group of inputs contributes to the variance of model output. Two of the most commonly used variance-based methods were applied in our study: *Sobol approach* and *Fourier Amplitude Sensitivity Test (FAST)*. These were implemented by using Monte Carlo algorithms and SIMLAB software tool for global sensitivity analysis [9]. Later the

development of a new Monte Carlo methods had been developed [22]. It combines properties of two famous approaches – Sobol quasi-Monte Carlo integration and a pseudorandom number SIMD-oriented Fast Mersenne Twister (SFMT) generator. The original quasirandom point is shaken a little bit, which leads to a more efficient numerical integration of the smooth integrands of low dimension [10].

Sensitivity studies are nowadays applied to some of the most complicated mathematical models from various intesively developing areas of application [3, 10, 19]. Different efficient stochastic algorithms for multidimensional integration have also been applied on a further stage of these sensitivity studies. The two Adaptive Monte Carlo algorithms that we are going to used are based on the ideas and results of the importance separation and are completely described in the following works [4, 5, 11, 12].

UNI-DEM is a powerful large-scale air pollution model for calculation of the concentrations of a large number of pollutants and other chemical species in the air along a certain time period. Its results can be used in various application areas (environmental protection, agriculture, health care, etc.). The large computational domain covers completely the European region and the Mediterranean.

2 Description of UNI-DEM

UNI-DEM is a powerful large-scale air pollution model for calculation of the concentrations of a large number of pollutants and other chemical species in the air along a certain time period. Its results can be used in various application areas (environmental protection, agriculture, health care, etc.). The large computational domain covers completely the European region and the Mediterranean.

The Unified Danish Eulerian Model (UNI-DEM) [42, 43, 45] is a complex and powerful large-scale air pollution for calculating the concentrations of a large number of pollutants and other chemical species in the air and their variation along certain time period (month, year, decade, etc.). The concentrations (and other application-oriented functions dependent on them) are calculated for a large geographical region (4800×4800 km), covering the whole of Europe, the Mediterranean and some parts of Asia and Africa. It takes into account the main physical, chemical and photochemical processes in the atmosphere. Large input data sets are needed for the emissions and the quickly changing meteorological conditions, which must also be taken into account. From the output results one can derive important information for environmental protection, agriculture, health care, etc.

UNI-DEM is mathematically represented by the following system of partial differential equations (PDE), in which the unknown concentrations c_s of a large number of chemical species (pollutants and other chemically active components) take part. The main physical and chemical processes (advection, diffusion, chemical reactions, emissions and deposition) are represented in that system.

$$\frac{\partial c_s}{\partial t} = -\frac{\partial (uc_s)}{\partial x} - \frac{\partial (vc_s)}{\partial y} - \frac{\partial (wc_s)}{\partial z} + \\ + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right) + \\ + E_s + Q_s(c_1, c_2, \ldots c_q) - (k_{1s} + k_{2s})c_s, \quad s = 1, 2, \ldots q \ . \tag{1}$$

where c_s are the concentrations of the chemical species; $u,\ v,\ w$ are the wind components along the coordinate axes; $K_x,\ K_y,\ K_z$—the diffusion coefficients; E_s—the emissions; $k_{1s},\ k_{2s}$—dry/wet deposition coefficients; $Q_s(c_1, c_2, \ldots c_q)$—non-linear functions describing the chemical reactions between the species under consideration.

The above PDE system is non-linear and stiff. Both non-linearity and stiffness are introduced mainly by the chemical scheme: the condensed CBM-IV (Carbon Bond Mechanism) [42, 43]. It is quite detailed and accurate, but computationally expensive as well.

The above PDE system is rather complex and not suitable for direct numerical treatment. It is non-linear. The chemical part formally presented by the term $Q_s(c_1, c_2, \ldots c_q)$ leads to a system of stiff ordinary differential equations [43]. Both non-linearity and stiffness are responsible for numerical and computational difficulties arising after some kind of discretization. The difficulties and challenges appear even when advanced numerical schemes are used (see, for instance [43]). For the purpose of its efficient numerical solution, (1) it is split into submodels, which represent the main physical and chemical processes [1].

The sequential splitting is used in the production version of the model, although other splitting methods have also been considered and implemented in some experimental versions [44].

3 Implementation of UNI-DEM

For the purpose of efficient numerical treatment, the system (1) is split according to the major physical and chemical processes and the following 3 submodels are formed: ***Advection-diffusion***, ***Chemistry & deposition*** and ***Vertical transport (vertical wind and convection)***.

The following methods are used in the numerical solution of the submodels:

- *Advection-diffusion part:* Finite elements, followed by predictor-corrector schemes with several different correctors.
- *Chemistry-deposition part:* An improved version of the QSSA (Quasi Steady-State Approximation) [18].
- *Vertical transport:* Finite elements, followed by theta-methods.

Spatial and time discretization makes each of the submodels a tough computational task even for the most advanced supercomputer systems. Efficient parallelization has always been a crucial point in the computer implementation of UNI-DEM. The task

Table 1 Time (T) and speed-up of SA-DEM code (MPI only) on the Spanish supercomputer IBM MareNostrum III at BSC, Barcelona

# CPU	# nodes	Advection		Chemistry		TOTAL		
		T [s]	Sp	T [s]	Sp	T [s]	Sp	E [%]
10	1	83469	10	76988	10	169872	10	100
40	3	20875	40	17382	44	40865	42	104
80	5	10823	77	9142	84	21971	77	97
160	10	5624	148	4625	166	12385	137	86
320	20	2889	288	2396	321	8165	208	65
640	40	1565	533	1192	646	5117	332	52
960	60	1252	667	818	941	3925	433	45
1600	100	816	1023	495	1555	2894	587	37

became much more challenging with development of the sensitivity analysis version of the code, SA-DEM [6, 7]. It consists of the following three parts:

- A modification of UNI-DEM with ability to modify certain parameters, subject to SA study. By now we have been interested in some chemical rate constants as well as in the input data for the anthropogenic emissions. A small number of input parameters is reserved for this purpose.
- A driver routine that automatically generates a set of tasks to produce the necessary results for a particular SA study. It allows to perform in parallel a large number of runs with common input data (reusing it), producing at once a whole set of values on a regular mesh (used later for calculating the sensitivity indices).
- An additional program for extracting the necessary mean monthly concentrations and computing the normalised ratios (to be analysed further on).

Significant improvements of the earlier versions of SA-DEM were made by introducing two additional levels of parallelism: top-level (MPI) and bottom-level (OpenMP) [27–29]. They allow us to use efficiently the computational power of the contemporary cluster supercomputers with multicore nodes. Additional improvements in the data management strategy, reducing the number of I/O operations and pipelining most of them with the computationally intensive stages, have also been done.

In Table 1 we show some scalability results from experiments with SA-DEM on one of the largest supercomputers in Europe—IBM MareNostrum III (in BSC, Barcelona, Spain). It consists of 3028 nodes IBM dx360 M4 (16 core) with 32 GB RAM per node. It is seen from Table 1 that the chemical stage (the most computationally expensive) scales very well (shows almost linear speed-up in the whole range of experiments). Advection stage scales pretty well in most of the experiments, with unerstandable slow-down in the highly parallel experiments. It is caused by the significant boundary overlapping of the domain partitioning when approaching the inherent partitioning limitations. In general, SA-DEM performs quite efficiently and show relatively high scalability on such a large computing system [27–29].

4 Sobol Approach for Global Sensitivity Indices

When the sensitivity of the concentrations calculated by UNI-DEM (or any other deterministic mathematical model) is studied, it is convenient to introduce some stochastic variables and equations.

Let s is the dimensionality of the problem under consideration. It is assume d that the mathematical model can be represented by a model function $\mathrm{u} = f(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_s) \in U^s \equiv [0; 1]^s$ is a vector of input parameters with joint probability density $p(\mathbf{x}) = p(x_1, \ldots, x_s)$.

The basic indicator corresponding to an input parameter x_i, $\imath = 1, \ldots, s$ (normalized between 0 and 1) is defined as $\dfrac{\mathbf{D}[\mathbf{E}[\mathrm{u}|x_i]]}{\mathbf{D}_\mathrm{u}}$, where $\mathbf{D}[\mathbf{E}[\mathrm{u}|x_i]]$ is the variance of the conditional mathematical expectation of u on x_i and $\mathbf{D}_\mathrm{u}$ is the total variance on u.

Total Sensitivity Index [20] is a measure of an input parameter's overall impact (total effect) and all the interactions involving that parameter.

Definition 1 The total sensitivity index of the input parameter x_i, $i \in \{1, \ldots, s\}$ is defined as [20, 35]:

$$S_{x_i}^{tot} = S_i + \sum_{l_1 \neq i} S_{il_1} + \sum_{l_1, l_2 \neq i, l_1 < l_2} S_{il_1l_2} + \ldots + S_{il_1 \ldots l_{s-1}}, \tag{2}$$

where S_i is called *main effect(first order sensitivity index)* of x_i and $S_{il_1 \ldots l_{j-1}}$ is the sensitivity index of j-th row (respectively *two-way interactions* for $j = 2$, *three-way interactions* for $j = 3$, etc.) for the parameter x_i $(2 \leq j \leq s)$.

The Sobol method is one of the most commonly used variance-based methods. The sensitivity measures, defined by Sobol [35], was first published in [36]. One significant advantage of this method is that it not only calculates first order indices but also high order sensitivity indices, and the total sensitivity index can be calculated with only one integral (one subintegral function) for a given parameter by a stochastic approach.

The Sobol method [35, 36] of global sensitivity analysis applied here is based on the decomposition of the integrable model function f (in s -dimensional space from input parameters) to terms with increasing dimensionality:

$$f(\mathbf{x}) = f_0 + \sum_{\nu=1}^{s} \sum_{l_1 < \ldots < l_\nu} f_{l_1 \ldots l_\nu}(x_{l_1}, x_{l_2}, \ldots, x_{l_\nu}), \tag{3}$$

where f_0 is a constant. The total number of terms in the Eq. (3) is 2^s (see [38]).

If each of the terms is subject to the following condition

$$\int_0^1 f_{l_1 \ldots l_\nu}(x_{l_1}, x_{l_2}, \ldots, x_{l_\nu}) \mathrm{d}x_{l_k} = 0, \quad 1 \leq k \leq \nu, \quad \nu = 1, \ldots, s, \tag{4}$$

then the representation (3) is unique and named ANOVA representation (**an**alysis **o**f **va**riance, analysis of variance) of the model function $f(\mathbf{x})$ [37].

Definition 2 Quantities

$$\mathbf{D} = \int_{U^s} f^2(\mathbf{x})\mathrm{d}\mathbf{x} - f_0^2, \quad \mathbf{D}_{l_1 \ldots l_\nu} = \int_{U^\nu} f^2_{l_1 \ldots l_\nu} \mathrm{d}x_{l_1} \ldots \mathrm{d}x_{l_\nu} \tag{5}$$

are called variances (total and partial variances, respectively) and are obtained after squaring and integrating over U^s with Eq. (3) assuming f(x) has a summable square.

Hence the total variance of the model output parameter is decomposed into the partial variances [36] in a similar way to the decomposition of the model function:

$$\mathbf{D} = \sum_{\nu=1}^{s} \sum_{l_1 < \ldots < l_\nu} \mathbf{D}_{l_1 \ldots l_\nu}, \tag{6}$$

which is the unique representation of the ANOVA type. Given the above assumptions about the model function and the variance of the output parameter:

Definition 3 ([36, 37]) The quantities

$$S_{l_1 \ldots l_\nu} = \frac{\mathbf{D}_{l_1 \ldots l_\nu}}{\mathbf{D}}, \quad \nu \in \{1, \ldots, s\} \tag{7}$$

are called global Sobol sensitivity indices.

Calculating the integrals given by the formulas (5) requires the integration of various sub-integer functions, which is not efficient from a computational point of view. Sobol has proposed a [37] computational procedure for calculating global sensitivity indices (in random order) that overcomes this drawback. The idea is based on arbitrary set of m variables ($1 \le m \le s-1$) defined as: $\mathrm{y} = (x_{k_1}, \ldots, x_{k_m})$, $1 \le k_1 < \ldots < k_m \le s$, and let z be the set of other $s-m$ variables. Hence $\mathbf{x} = (\mathrm{y}, \mathrm{z})$. Let $K = (k_1, \ldots, k_m)$.

The variances corresponding to the sets of variables y and z are defined as

$$\mathbf{D}_\mathrm{y} = \sum_{n=1}^{m} \sum_{(i_1 < \ldots < i_n) \in K} \mathbf{D}_{i_1 \ldots i_n}, \quad \mathbf{D}_\mathrm{z} = \sum_{n=1}^{s-m} \sum_{(j_1 < \ldots < j_n) \in \bar{K}} \mathbf{D}_{j_1 \ldots j_n}, \tag{8}$$

where the complement of the subset K to the set of indices of all input parameters is indicated by $\bar{K}$. The first sum in (8) is plotted on all subsets $(i_1, \ldots, i_n)$, where all subscripts $i_1, \ldots, i_n$ belong to K. Then the total variance of the subset y is $\mathbf{D}_\mathrm{y}^{tot} = \mathbf{D} - \mathbf{D}_\mathrm{z}$ and is written over all subsets $(i_1, \ldots, i_\nu)$, $1 \le \nu \le s$, where at least one index belongs to K: $i_l \in K$, $1 \le l \le \nu$. The procedure for computation of global sensitivity indices is based on the following variance representation $\mathbf{D}_\mathrm{y}$:

$$\mathbf{D}_{\mathrm{y}} = \int_{U^{2s-m}} f(\mathbf{x})\ f(\mathrm{y}, \mathrm{z}')\mathrm{d}\mathbf{x}\mathrm{d}\mathrm{z}' - f_0^2 \tag{9}$$

(see [37]).

Following the idea of Homma and Salteli in [20] better evaluation for the first-order terms are obtained if f_0^2 in (5) is approximated directly using the relation

$$f_0^2 = \int_{U^{2s}} f(\mathbf{x}) f(\mathbf{x}')\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{x}' \tag{10}$$

instead of $f_0^2 = \left[\int_{U^s} f(\mathbf{x})\mathrm{d}\mathbf{x}\right]^2$. This derives from the formula (9) where the corresponding estimate of first-order indices $S_{l_1}, l_1 \in \{1, \ldots, s\}$ goes more naturally to zero for a non-influential input x_{l_1} if (10) is used [31] and can be detected by the formula:

$$D_{l_1} = \int_{U^{2s}} f(\mathbf{x})\ [f(x_1', \ldots, x_{l_1-1}', x_{l_1}, x_{l_1+1}', \ldots, x_s') - f(x_1', \ldots, x_d')]\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{x}'.$$

Following the idea of Saltelli in [31] about the computation of the total sensitivity indices is to use the following estimate for $S_{l_1}^{tot}$:

$$1 - \frac{1}{\mathbf{D}}\left\{\int_{U^{s+1}} f(x_1, \ldots, x_s)\ f(x_1, \ldots, x_{l_1-1}, x_{l_1}', x_{l_1+1}, \ldots, x_s)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{x}_{l_1}' - f_0^2\right\},$$

where the evaluation of f_0 is based only on one of the total two independent samples.

The computational complexity of calculating all first-order sensitivity indices using the described Sobol approach is determined by $N(2s + 1)$ in the model values (N values of the function for f_0, sN first order index values, and sN total sensitivity index values), where N is the dimension of the sample and s is the number of input parameters. It should be noted that the most commonly used variance-based sensitivity analysis methods, such as Sobol and FAST (Fourier Amplitude Sensitivity Test), and their improved versions, are characterized by computational complexity proportional to sN for the estimation of all first-order indices and total sensitivity indices of the input parameters (see [31]).

That's why the mathematical treatment of the problem for multidimensional sensitivity analysis is contained in evaluating total sensitivity indices and in particular Sobol' sensitivity indices (7) of corresponding order which leads to computing of multidimensional integrals.

5 Optimized Latin Hypercube Sampling

Latin Hypercube Sampling is a type of stratified sampling, described for the first time by M.D. McKay in 1979 in [23]. For this standard method we will use the notation LHS. Recently a renewing interest in optimization version of Latin hypercube sampling is described by Budiman Minasny in his works [24, 25]. In the case of integral approximation we must simply divide the domain $[0, 1]^d$ into m^d disjoint subdomains, each of volume $\frac{1}{m^d}$ and to sample one point from each of them. Let this sample be $\mathbf{x}_{k,j}$, for dimensions $k = 1, \ldots, m^d$, $j = 1, \ldots, d$. LHS does not require more samples for more dimensions (variables)—it is one of the main advantages of this scheme. Examples of random, stratified and Latin hypercube samplings with 16 points are presented on Fig. 1 [21].

Here we propose an optimized version of the Latin Hypercube Sampling which includes a uniformly distributed random point in each interval which improves the accuracy as can be seen from the results below. The description of the optimized latin hypercube sampling (LHSO) algorithm is the following:

1. Divide the distribution of each variable X into n equiprobable intervals.
2. For each interval, choose one uniform random number $r \in [0, 1]$. Then, let

$$Prob_i = \frac{i-1}{n} + r * \frac{1}{n} \tag{11}$$

3. Transform the probability into the sampled value x using the inverse of the cumulative distribution function:

$$x = F^{-1}(Prob) \tag{12}$$

4. The n values obtained for each variable X are paired randomly (e.g. by generating a random permutation of $\{1, 2, \ldots, n\}$ for each variable except one) with the n values of the other variables.

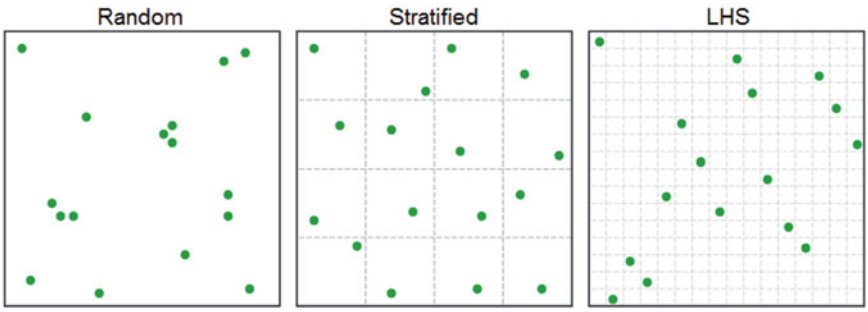


Fig. 1 Comparison of random, stratified and latin hypercube samplings with 16 points (d = 2, M = 4).

6 Sensitivity Studies with Respect to Emission Levels

The huge output data stream of UNI-DEM contains are the mean monthly concentrations of more than 30 pollutants. We consider 2 of them: *ozone* (O_3) and *ammonia* (NH_3). In particular, we present some results of a sensitivity study of the mean monthly concentrations of ammonia.

Here we present some results of our research on the sensitivity of UNI-DEM output (in particular, the ammonia mean monthly concentrations) with respect to the anthropogenic emissions input variation. The anthropogenic emissions input consists of 4 different components $\mathbf{E} = (\mathbf{E^A}, \mathbf{E^N}, \mathbf{E^S}, \mathbf{E^C})$ as follows:

$$\mathbf{E^A} - \text{ammonia } (NH_3);$$
$$\mathbf{E^N} - \text{nitrogen oxides } (NO + NO_2);$$

$$\mathbf{E^S} - \text{sulphur dioxide } (SO_2);$$
$$\mathbf{E^C} - \text{anthropogenic hydrocarbons.}$$

The domain is the 4-dimensional hypercube $[0.5, 1]^4$. The input data have been generated by the improved version SA-DEM code, specialized for sensitivity studies (see the previous section).

The results for relative errors for evaluation of the quantities f_0, total variances and first-order and total sensitivity indices using various stochastic approaches for numerical integration are presented in Tables 2, 3, 4, respectively. The quantity f_0 is presented by a 4-dimensional integral whereas the rest of the quantities under consideration are presented by an 8-dimensional integrals. In this work for the estimate quantity we use the notation EQ.

Two Adaptive approaches from our previous study have been used in our comparison [30]. The results show that the algorithms using LHSO simulate the behaviour of the Adaptive Monte Carlo algorithm, and for some small in value sensitivity indices as S_4^{tot} LHSO gives better results than the adaptive approach. Adaptive algorithm gives results of the same order as LHS, and sometimes even outperforms it (see for example the relative errors for S_1^{tot}).

Most influential emissions about ammonia output concentrations are ammonia emissions themselves (about 89% for Milan). The second most influential emissions about ammonia output are sulphur dioxide emissions (about 11%)—see Fig. 2. Pie

Table 2 Relative error for the evaluation of $f_0 \approx 0.048$

# of samples	ADAPT1	ADAPT2	LHSO	LHS
	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{16}	9.89e-05	4.05e-04	1.45e-05	7.74e-05
2^{18}	3.95e-05	3.83e-06	1.98e-06	3.80e-06
2^{20}	4.99e-05	2.93e-05	6.31e-06	7.16e-06

Table 3 Relative error for the evaluation of the total variance $\mathbf{D} \approx 0.0002$

# of samples	ADAPT1	ADAPT2	LHSO	LHS
	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{16}	1.83e-04	5.12e-04	2.17e-04	3.65e-04
2^{18}	5.77e-05	1.21e-04	4.56e-05	1.21e-05
2^{20}	3.42e-05	3.28e-05	1.87e-05	5.96e-05

Table 4 Relative error for estimation of sensitivity indices of the input anthropogenic emissions by using various Monte Carlo approaches ($n = 2^{16} = 65536$)

EQ	Ref. value	ADAPT1	ADAPT2	LHSO	LHS
S_1	9**e**-01	7.67e-04	1.22e-03	1.16e-04	9.79e-03
S_2	2**e**-04	1.47e-03	4.96e-02	7.10e-03	6.60e-01
S_3	1**e**-01	4.11e-03	1.59e-03	1.19e-03	8.65e-03
S_4	4**e**-05	1.04e-01	1.69e-01	6.10e-02	6.70e-01
S_1^{tot}	9**e**-01	4.99e-05	5.36e-05	7.31e-05	4.31e-04
S_2^{tot}	2**e**-04	5.23e-01	5.00e+00	1.07e-01	2.94e+01
S_3^{tot}	1**e**-01	1.15e-02	1.28e-02	5.45e-03	1.10e-02
S_4^{tot}	5**e**-05	1.88e+01	3.43e+01	7.54e-01	2.41e+02

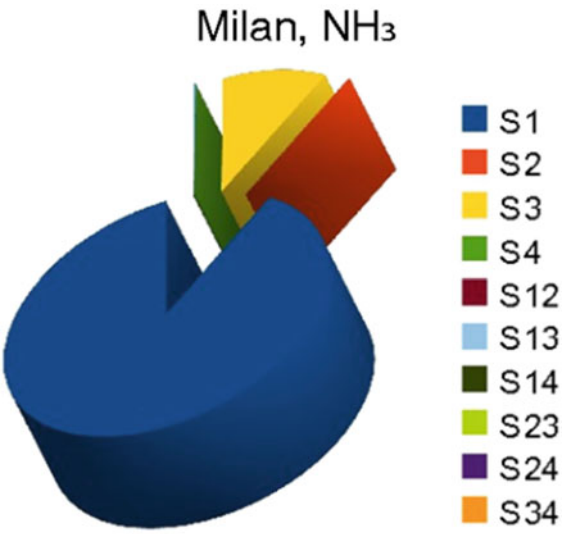


Fig. 2 Pie charts representation of first- and second-order sensitivity indices of the ammonia in Milan

charts representation of first- and second-order sensitivity indices of the ammonia in Milan on Fig. 2 has been obtained applying the correlated sampling of Sobol' variance-based approach for multidimensional sensitivity analysis for computing all possible sensitivity measures to study influence of four chosen groups of air pollutant emissions over the concentration of the three important air pollutants.

7 Sensitivity Studies with Respect to Chemical Reactions Rates

We will also study the sensitivity of the ozone concentration values in the air over Genova with respect to the rate of variation of some chemical reactions of the condensed CBM-IV scheme [42], namely: # 1, 3, 7, 22 (time-dependent) and # 27, 28 (time independent). The simplified chemical equations of those reactions are as follows:

$$
\begin{aligned}
&[\#\mathbf{1}]\ \ NO_2 + h\nu \Longrightarrow NO + O;\\
&[\#\mathbf{3}]\ \ O_3 + NO \Longrightarrow NO_2;\\
&[\#\mathbf{7}]\ \ NO_2 + O_3 \Longrightarrow NO_3;
\end{aligned}
$$

$$
\begin{aligned}
&[\#\mathbf{22}]\ \ HO_2 + NO \Longrightarrow OH + NO_2;\\
&[\#\mathbf{27}]\ \ HO_2 + HO_2 \Longrightarrow H_2O_2;\\
&[\#\mathbf{28}]\ \ OH + CO \Longrightarrow HO_2.
\end{aligned}
$$

The domain under consideration is the 6-dimensional hypercube $[0.6, 1.4]^6$.

Homma and Saltelli discuss in [20] which is the better estimation of $f_0^2 = \left(\int_{U^d} f(\mathbf{x})\mathrm{d}\mathbf{x}\right)^2$ in the expression for total variance and Sobol global sensitivity measures. In case of estimating sensitivity indices of a fixed order, the formula $f_0^2 \approx \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,d})\ f(\mathbf{x}'_{i,1}, \ldots, \mathbf{x}'_{i,d})$, where $\mathbf{x}$ and $\mathbf{x}'$ are two independent sample vectors, is better (as recommended in [20, 40]).

The relative errors for evaluation of the quantities f_0, total variances, first and second order sensitivity indices by using various stochastic approaches for numerical integration are presented in Tables 5, 6, 7 respectively. Here the quantity f_0 is presented by a 6-dimensional integral, whereas the total variance and the sensitivity indices are presented by 12-dimensional integrals, following the ideas of *correlated sampling*.

We can conclude that all stochastic approaches under consideration give reliable relative errors for sufficiently large number of samples. The adaptive MC algorithm outperforms LHS in the case of small number of samples, higher dimensions and small by value sensitivity indices. The algorithms using generalized LHSO simulate the behaviour of Adaptive Monte Carlo algorithm, and even for higher dimensions

Table 5 Relative error for evaluation of $f_0 \approx 0.27$

# of samples	ADAPT1	ADAPT2	LHSR	LHS
	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{16}	3.49e-05	3.72e-05	1.12e-05	2.02e-04
2^{18}	5.90e-07	1.02e-06	6.14e-06	2.82e-05
2^{20}	1.36e-07	5.56e-07	8.71e-07	1.04e-05

Table 6 Relative error for evaluation of the total variance $\mathbf{D} \approx 0.0025$

# of samples	ADAPT1	ADAPT2	LHSO	LHS
	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{16}	3.01e-05	1.48e-04	8.23e-05	3.56e-05
2^{18}	7.71e-06	3.79e-04	6.12e-05	7.78e-03
2^{20}	1.75e-06	3.34e-05	5.21e-05	2.78e-04

Table 7 Relative error for estimation of sensitivity indices of several chemical reaction rate parameters by using various Monte Carlo and quasi-Monte Carlo approaches ($n = 2^{16} = 65536$)

EQ	Ref. value	ADAPT1	ADAPT2	LHSO	LHS
S_1	4**e**-01	1.55e-04	3.48e-04	7.53e-05	3.04e-02
S_2	3**e**-01	4.34e-04	1.58e-04	5.19e-04	7.35e-04
S_3	5**e**-02	3.42e-04	8.09e-05	7.24e-04	2.33e-02
S_4	3**e**-01	4.75e-04	9.04e-04	4.31e-04	2.47e-02
S_5	4**e**-07	1.31e+01	1.07e+01	2.37e+01	9.25e+02
S_6	2**e**-02	1.08e-03	4.54e-04	5.21e-04	3.81e-02
S_{12}	6**e**-03	1.30e-02	7.92e-03	7.70e-03	8.99e-02
S_{14}	5**e**-03	5.30e-03	1.81e-03	5.11e-03	2.74e-01
S_{15}	8**e**-06	9.34e+02	9.34e+02	9.06e+02	9.21e+02
S_{24}	3**e**-03	1.26e-03	7.24e-03	4.21e-03	7.10e-01
S_{45}	1**e**-05	9.93e-02	8.55e-02	1.14e-01	1.05e+01

sometimes gives better results than the to adaptive approach by at least one order. The most efficient in terms of computational complexity is the LHSO algorithm, followed very closely by the LHS algorithm. The algorithm LHS is characterized with unreliable relative errors for small in value sensitivity measures. MC approach based on optimized Latin hypercube sampling produce significantly better results for 6-dimensional and 12-dimensional integrals in comparison with the standard latin hypercube sampling LHS.

The representation of the reference values of the first-order and second-order sensitivity indices of input parameters (for ozone concentrations) are given on Fig. 3.

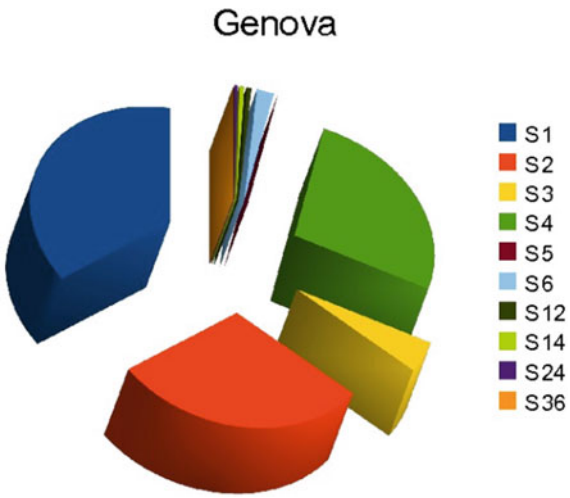


Fig. 3 Pie charts representation of first- and second-order sensitivity indices of the ozone in Genova

One can observe that the second-order sensitivity indices take rather small portions. This fact shows that the UNI-DEM is additive according to the rates of the six chemical reactions studied in this work.

8 Conclusion

The present study focuses on the so-called environmental safety. The computational efficiency in terms of relative error of eight stochastic algorithms for multidimensional numerical integration has been studied to analyze the sensitivity of UNI-DEM model output to variation of input emissions of the anthropogenic pollutants and of rate constants of selected chemical reactions. The algorithms have been successfully applied to compute global Sobol sensitivity measures corresponding to the influence of several input parameters on the concentrations of important air pollutants.

A comprehensive experimental study of Monte Carlo algorithm based on optimized latin hypercube sampling and adaptove approach for multidimensional numerical integration has been done. Such comparison has been made for the first time and this motivates the present study. The algorithms have been successfully applied to compute global Sobol sensitivity measures corresponding to the six chemical reactions rates and four different groups of pollutants. The numerical tests show that the optimized stochastic approach gives comparable results to the adaptive approach and better than the standard latin hypercube sampling and this method is very efficient for multidimensional numerical integration.

Sensitivity analysis, and in particular the results obtained, play an extremely important two-sided role: to test and improve mathematical models, and, on the other hand, to reliably interpret the numerical results by relevant specialists. Variance-

based analysis is a very effective tool for in-depth study of the relationship between individual parameters, outputs, and internal mechanisms governing the system in question. By identifying the major chemical reactions that affect the behavior of the system, specialists in various fields of application (physics, chemistry) will be able to obtain valuable information about improving the model, which in turn will increase the reliability and sustainability of forecasts. Thus, through a sensitivity analysis, the mathematical model will help to make more accurate estimates of the effects of harmful emissions on human health and losses in agriculture.

Acknowledgements Venelin Todorov is supported by the Bulgarian National Science Fund under Project KP-06-M32/2-17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics" and by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No D01-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria. This work is supported by the Bulgarian National Science Fund under Project DN 12/5-2017 and Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications" funded by National Science Fund—Bulgaria. Barcelona Supercomputing Centre (BSC) is kindly acknowledged too for granting us access and computer time on their most powerful supercomputer.

References

1. Csomós, P., Faragó, I., Havasi, A.: Weighted sequential splittings and their analysis. Comp. Math. Appl. **50**, 1017–1031 (2005)
2. Cukier, R., Fortuin, C., Shuler, K., Petschek, A., Schaibly, J.: Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I. Theory. J. Chem. Phys. **59**, 3873–3878 (1973)
3. Dimitriu, G.: Global sensitivity analysis for a chronic myelogenous leukemia model. In: Proceedings of the 9th International Conference NMA'2018, Borovets, Bulgaria, August 20–24, 2018, LNCS 11189. Springer (2019). https://doi.org/10.1007/978-3-030-10692-8_42
4. Dimov, I.T., Georgieva, R.: Monte Carlo method for numerical integration based on sobol' sequences. In: Dimov, I., Dimova, S., Kolkovska, N. (eds.) Numerical Methods and Applications, LNCS 6046, pp. 50–59. Springer (2011)
5. Dimov, I.T., Georgieva, R.: Multidimensional sensitivity analysis of large-scale mathematical models. In: Iliev, O.P., et al. (eds.), Numerical Solution of Partial Differential Equations: Theory, Algorithms, and Their Applications, Springer Proceedings in Mathematics & Statistics 45, pp. 137–156. Springer Science+Business Media, New York (2013)
6. Dimov, I.T., Georgieva, R., Ostromsky, T., Zlatev, Z.: Variance-based sensitivity analysis of the unified danish eulerian model according to variations of chemical rates. In: Dimov, I., Faragó, I., Vulkov, L., (eds.) Proceedings of the NAA 2012, LNCS 8236, pp. 247–254. Springer (2013)
7. Dimov, I.T., Georgieva, R., Ostromsky, T., Zlatev, Z.: Sensitivity studies of pollutant concentrations calculated by UNI-DEM with respect to the input emissions. Cent. Eur. J. Math., "Numerical Methods Large Scale Sci. Comput." **11**(8), 1531–1545 (2013)
8. Dimov, I.T., Georgieva, R., Ostromsky, T., Zlatev, Z.: Advanced algorithms for multidimensional sensitivity studies of large-scale air pollution models based on sobol sequences. Spec. Issue Comput. Math. Appl. **65**(3); "Efficient Numerical Methods for Scientific Applications", pp. 338–351. Elsevier (2013)

9. Dimov, I., Georgieva, R., Ostromsky, T.: Monte Carlo sensitivity analysis of an Eulerian large-scale air pollution model. Reliab. Eng. Syst. Saf. **107**, 23–28 (2012). Elsevier
10. Dimov, I., Georgieva, R.: Monte Carlo algorithms for evaluating Sobol' sensitivity indices. Math. Comput. Simul. **81**(3), 506–514 (2010)
11. Dimov, I., Karaivanova, A.: Error analysis of an adaptive Monte Carlo method for numerical integration. Math. Comput. Simul. **47**, 201–213 (1998)
12. Dimov, I., Karaivanova, A., Georgieva, R., Ivanovska, S.: Parallel Importance Separation and Adaptive Monte Carlo Algorithms for Multiple Integrals. Springer Lecture Notes in Computer Science, pp. 99–107. Springer, Berlin, Heidelberg, New York (2003)
13. Ferretti, F., Saltelli A., Tarantola, S.: Trends in sensitivity analysis practice in the last decade. J., Sci. Total. Environ. Special issue on Human and Biota Exposure **568**, 666–670 (2016). Elsevier
14. Georgiev, I.R., Zheleva, I., Filipova, M.: Numerical study of the influence of two-burner heating upon the heat transfer during pyrolysis process used for end-of-life tires (EOLT) treatment. In: AIP Conference Proceedings, vol. 2164
15. Gocheva-Ilieva, Snezhana G., Ivanov, Atanas V., Livieris, Ioannis E.: High performance machine learning models of large scale air pollution data in urban area. Cybern. Inf. Technol. **20**(6), 49–60 (2020)
16. Gocheva-Ilieva, S.G., Ivanov, A.V., Voynikova, D.S., et al.: Time series analysis and forecasting for air pollution in small urban area: an SARIMA and factor analysis approach. Stoch Environ Res Risk Assess **28**, 1045–1060 (2014)
17. Gocheva-Ilieva, S.G., Voynikova, D.S., Stoimenova, M.P., Ivanov, A.V., Iliev, I.P.: Regression trees modeling of time series for air pollution analysis and forecasting. Neural Comput. Appl. **31**(12), 9023–9039 (2019)
18. Hesstvedt, E., Hov, Ø., Isaksen, I.A.: Quasi-steady-state approximations in air pollution modeling: comparison of two numerical schemes for oxidant prediction. Int. J. Chem. Kinet. **10**, 971–994 (1978)
19. Hamdad, H., Pézerat, C., Gauvreau, B., Locqueteau, C., Denoual, Y.: Sensitivity analysis and propagation of uncertainty for the simulation of vehicle pass-by noise. Appl. Acoust. **149**, 85-98 (2019) Elsevier. https://doi.org/10.1016/j.apacoust.2019.01.026
20. Homma, T., Saltelli, A.: Importance measures in global sensitivity analysis of nonlinear models. Reliab. Eng. Syst. Saf. **52**, 1–17 (1996)
21. Jarosz, W.: Efficient Monte Carlo methods for light transport in scattering media, Ph.D. dissertation, UCSD (2008)
22. Kroese, D.P., Taimre, T., Botev, Z.: Handbook of Monte Carlo Methods. Wiley Series in Probability and Statistics (2011)
23. McKay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**(2), 239–45 (1979)
24. Minasny, B., McBratney, B.: A conditioned Latin hypercube method for sampling in the presence of ancillary information. J. Comput. Geosci. Arch. **32**(9), 1378–1388 (2006)
25. Minasny, B., McBratney, B.: Conditioned Latin hypercube sampling for calibrating soil sensor data to soil properties. In: Chapter: Proximal Soil Sensing, Progress in Soil Science, pp. 111–119 (2010)
26. Ökten, G., Göncüb, A.: Generating low-discrepancy sequences from the normal distribution: Box-Muller or inverse transform? Math. Comput. Model. **53**, 1268–1281 (2011)
27. Ostromsky, T., Dimov, I.T., Georgieva, R., Zlatev, Z.: Parallel computation of sensitivity analysis data for the Danish Eulerian model. In: Proceedings of the 8th International Conference LSSC'11, LNCS-7116, pp. 301–309. Springer (2012)
28. Ostromsky, T., Dimov, I.T., Georgieva, R., Zlatev, Z.: Air pollution modelling, sensitivity analysis and parallel implementation. Int. J. Environ. Pollut. **46**(1/2), 83–96 (2011)
29. Ostromsky, T., Dimov, I.T., Marinov, P., Georgieva, R., Zlatev, Z.: Advanced sensitivity analysis of the Danish Eulerian Model in parallel and grid environment. In: Proceedings of the 3rd International Conference AMiTaNS'11, AIP Conference Proceedings, vol. 1404, pp. 225–232 (2011)

30. Ostromsky, T., Todorov, V., Dimov, I., Zlatev, Z.: Sensitivity studies of an air pollution model by using efficient stochastic algorithms for multidimensional numerical integration. In: Dimov, I., Fidanova, S., (eds.) Advances in High Performance Computing. HPC 2019. Studies in Computational Intelligence, vol. 902. Springer, Cham (2019)
31. Saltelli, A.: Making best use of model valuations to compute sensitivity indices. Comput. Phys. Commun. **145**, 280–297 (2002)
32. Saltelli, A., Chan, K., Scott, M.: Sensitivity Analysis. Probability and Statistics series, Wiley (2000)
33. Saltelli, A., Tarantola, S., Campolongo, F., Ratto, M.: Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models. Halsted Press, New York (2004)
34. Poryazov, S., Saranova, E., Ganchev, I.: Conceptual and Analytical Models for Predicting the Quality of Service of Overall Telecommunication Systems. Autonomous Control for a Reliable Internet of Services, pp. 151–181. Springer, Cham (2018)
35. Sobol', I.M.: Sensitivity estimates for nonlinear mathematical models. Math. Model. Comput. Exp. **1**, 407–414 (1993)
36. Sobol', I.M.: Sensitivity estimates for nonlinear mathematical models. Matem. Model. **2**(1), 112–118 (1990)
37. Sobol', I.M.: Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math. Comput. Simul. **55**(1–3), 271–280 (2001)
38. Sobol', I.M.: Theorem and examples on high dimensional model representation. Reliab. Eng. Syst. Saf. **79**, 187–193 (2003)
39. Sobol, I.M., Myshetskaya, E.: Monte Carlo estimators for small sensitivity indices. Monte Carlo Methods Appl. **13**(5–6), 455–465 (2007)
40. Sobol, I.M., Tarantola, S., Gatelli, D., Kucherenko, S., Mauntz, W.: Estimating the approximation error when fixing unessential factors in global sensitivity analysis. Reliab. Eng. Syst. Saf. **92**, 957–960 (2007)
41. Zheleva, I., Georgiev, I., Filipova, M., Menseidov, D.: Mathematical modeling of the heat transfer during pyrolysis process used for end-of-life tires treatment. In: AIP Conference Proceedings, vol. 1895, no. 1, p. 030008. AIP Publishing LLC
42. Zlatev, Z.: Computer Treatment of Large Air Pollution Models. KLUWER Academic Publishers, Dorsrecht-Boston-London (1995)
43. Zlatev, Z., Dimov, I.T.: Computational and Numerical Challenges in Environmental Modelling. Elsevier, Amsterdam (2006)
44. Zlatev, Z., Dimov, I., Georgiev, K.: Modeling the long-range transport of air pollutants. IEEE Comput. Sci. Eng. **1**(3), 45–52 (1994)
45. Zlatev, Z., Dimov, I., Georgiev, K.: Three-dimensional version of the Danish Eulerian model. Zeitschrift für Angewandte Mathematik und Mechanik **76**(S4), 473–476 (1996)

Optimized Quasi-Monte Carlo Methods Based on Van der Corput Sequence for Sensitivity Analysis in Air Pollution Modelling

Venelin Todorov, Ivan Dimov, Tzvetan Ostromsky, Zahari Zlatev, Rayna Georgieva, and Stoyan Poryazov

Abstract An optimization version of the van der Corput sequence has been used in our sensitivity studies of the model output results for some air pollutants with respect to the emission levels and some chemical reactions rates. Sensitivity analysis of model outputs to variation or natural uncertainties of model inputs is very significant for improving the reliability of these models. Clearly, the progress in the area of air pollution modeling, is closely connected with the progress in reliable algorithms for multidimensional integration.

Keywords Quasi-Monte Carlo methods · Air pollution model · Sensitivity analysis

V. Todorov · S. Poryazov
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
e-mail: stoyan@math.bas.bg

V. Todorov (✉) · I. Dimov · T. Ostromsky · R. Georgieva
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

I. Dimov
e-mail: ivdimov@bas.bg

T. Ostromsky
e-mail: ceco@parallel.bas.bg

R. Georgieva
e-mail: rayna@parallel.bas.bg

Z. Zlatev
National Centre for Environment and Energy, University of Århus, RoskildeFrederiksborgvej 399, P.O. Box 358, 4000 Roskilde, Denmark
e-mail: zz@dmu.dk

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_20

1 Introducton

Mathematical models are used to study and predict the behavior of a variety of complex systems—engineering, physical, economic, social, environmental [15, 28, 42, 44]. They determine the most important quantities that control the state and behavior of a system, as well as the quantitative regularities, that is, the mathematical laws that underlie the change of these quantities. On the other hand, it is of significant importance to develop techniques to determine the reliability, robustness and efficiency of a mathematical model. One such approach is sensitivity analysis (SA). Sensitivity studies are nowadays applied to some of the most complicated mathematical models from various intensively developing areas of application [5, 6, 16–19].

Usually, the computational problems arising from the use of large mathematical models of long-range transport of pollutants in the air are extremely laborious. However, there are serious computational challenges, even when using advanced high-performance computer systems [47]. Therefore, it would be good to simplify the model while maintaining the necessary order of accuracy and reliability of the model results. Therefore, it is necessary to perform a precise SA to determine where and how simplifications can be made. On the other hand, it is extremely important to carefully analyze the effect of changes in initial conditions, boundary conditions and/or rate constants of chemical reactions on the model results in order to make correct assumptions about possible simplifications. SA will help for model improvement, increase the reliability of results, and identify the parameters and mechanisms that need to be examined more precisely because the model results are very sensitive to their variation.

An important issue when large-scale mathematical models are used to support decision makers is their reliability. Sensitivity analysis of model outputs to variation

Table 1 Relative error for the evaluation of $f_0 \approx 0.048$

	VDC2	VDCO	FIBO	LATM
# of samples	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{10}	3.72e-04	1.64e-05	2.09e-04	8.46e-04
2^{12}	4.39e-04	5.11e-05	4.32e-05	1.79e-04
2^{14}	2.15e-03	4.45e-05	2.25e-05	2.62e-06
2^{16}	3.11e-04	5.56e-06	8.70e-06	4.14e-07
2^{18}	7.66e-05	1.03e-06	1.79e-06	1.17e-06
2^{20}	1.48e-04	3.56e-07	4.21e-07	1.15e-06

Table 2 Relative error for the evaluation of the total variance $\mathbf{D} \approx 0.0002$

	VDC2	VDCO	FIBO	LATM
# of samples	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{10}	9.21e-02	1.78e-02	1.63e-01	1.54e-02
2^{12}	6.37e-02	4.11e-03	2.39e-02	3.67e-03
2^{14}	4.26e-02	1.34e-03	2.90e-03	1.49e-03
2^{16}	2.22e-03	3.19e-04	2.65e-04	1.61e-03
2^{18}	2.22e-03	1.13e-04	3.01e-04	1.48e-03
2^{20}	7.58e-03	5.76e-05	1.19e-04	1.46e-03

Table 3 Relative error for estimation of sensitivity indices of input parameters using various Monte Carlo and quasi-Monte Carlo approaches ($n \approx 65536$)

Est. qnt.	Ref. val.	VDC2	VDCO	FIBO	LATM
S_1	9**e**-01	3.13e-02	4.56e-04	3.62e-04	7.27e-04
S_2	2**e**-04	1.28e+00	3.34e-02	1.74e-01	2.76e-02
S_3	1**e**-01	9.13e-02	2.22e-03	3.22e-03	4.24e-03
S_4	4**e**-05	8.30e-01	3.45e-02	4.87e-01	1.65e-02
S_1^{tot}	9**e**-01	7.54e-03	1.16e-04	4.61e-04	5.14e-04
S_2^{tot}	2**e**-04	4.69e+01	1.24e-01	3.45e-01	2.21e-01
S_3^{tot}	1**e**-01	4.14e-02	1.10e-03	1.96e-03	6.41e-03
S_4^{tot}	5**e**-05	5.54e+02	1.60e-01	5.06e-01	1.60e-01

Table 4 Relative error for the evaluation of $f_0 \approx 0.27$

	VDC2	VDCO	FIBO	LATM
# of samples	Relative	Relative	Relative	Relative
n	Error	Error	Error	Error
2^{10}	1.25e-02	8.13e-04	2.08e-03	7.12e-03
2^{12}	1.70e-03	2.12e-04	1.40e-04	1.80e-03
2^{14}	3.56e-03	6.15e-05	3.98e-04	4.04e-05
2^{16}	8.66e-04	7.09e-05	2.61e-04	9.91e-06
2^{18}	4.63e-04	2.33e-06	7.29e-06	7.24e-06
2^{20}	7.85e-05	3.51e-07	4.57e-07	7.04e-06

Table 5 Relative error for the evaluation of the total variance $\mathbf{D} \approx 0.0025$

	VDC2	VDCO	FIBO	LATM
# of samples	Relative	Relative	Relative	Relative
n	Eerror	Error	Error	Error
2^{10}	2.23e-02	1.10e-02	6.73e+00	3.11e-02
2^{12}	2.04e-01	8.45e-03	5.27e-01	8.76e-02
2^{14}	3.94e-02	6.44e-04	1.02e-01	7.54e-04
2^{16}	1.05e-03	2.71e-04	1.97e-03	9.13e-04
2^{18}	1.27e-02	1.01e-04	4.53e-03	2.22e-03
2^{20}	1.53e-02	9.56e-05	9.33e-03	2.22e-03

or natural uncertainties of model inputs is very significant for improving the reliability of these models. In a popular definition (due to A. Saltelli [31]), sensitivity analysis (SA) is the study of how uncertainty in the output of a model can be apportioned to different sources of uncertainty in the model input. The uncertainty in the model input in our case, as in many others, can be due to various reasons: inaccurate measurements or calculation, approximation, data compression, etc. Two kinds of sensitivity analysis have been discussed in the literature: local and global [29, 30]. Local SA studies how much some small variations of inputs around a given value can change the value of the output. Global SA takes into account all the variation range of the input parameters, and apportions the output uncertainty to the uncertainty in the input data. In the current paper subject to our study is the global sensitivity analysis, applied to a specific large-scale air pollution model.

The main goal of the present work is to develop and investigate efficient stochastic algorithms for multiple numerical integration providing sensitivity analysis (SA) that means evaluating Sobol' sensitivity indices (SIs) [6, 9, 19, 28]. While the classical deterministic grid methods are efficient for low dimensional integrands [7], they become computationally intensive and even impracticable for high dimensions s because the number of required integrand evaluations grows exponentially. In contrast, the convergence rate of the plain **M**onte **C**arlo (MC) integration methods [7] does not depend on the number of dimensions s. That is why the Monte Carlo quasi Monte Carlo methods [1–4] is a power tool in sensitivity analysis of large-scale systems [13, 14].

The Unified Danish Eulerian Model (UNI-DEM) [45, 46] is in the focus of our investigation as one of the most advanced large-scale mathematical models that describes adequately all physical and chemical processes. UNI-DEM is a powerful large-scale air pollution model for calculation the concentrations of a large number of pollutants and other chemical species in the air. The calculations are done in a large spatial domain, which covers completely the European region and the Mediterranean, for certain time period (meteorological data must be available for it) [9]. In this particular study we use them for two of the most dangerous pollutants: the ozone (O_3) and the ammonia (NH_3). Other accumulative functions related to sone specific

Table 6 Relative error for estimation of sensitivity indices of input parameters using various QMC approaches ($n \approx 65536$)

Est. qnt.	Ref. val.	VDC2	VDCO	FIBO	LATM
S_1	4**e**-01	1.07e-01	1.92e-02	3.82e-02	1.50e-02
S_2	3**e**-01	5.08e-02	7.45e-03	1.03e-02	2.14e-02
S_3	5**e**-02	6.37e-04	5.22e-04	5.48e-01	8.28e-02
S_4	3**e**-01	7.39e-02	4.71e-03	1.07e-02	6.81e-03
S_5	4**e**-07	7.26e+02	9.00e+01	3.40e+03	2.07e+03
S_6	2**e**-02	4.10e-01	5.56e-03	1.32e+00	1.19e-02
S_1^{tot}	4**e**-01	8.89e-02	6.56e-02	7.92e-02	1.07e-02
S_2^{tot}	3**e**-01	1.06e-02	1.41e-02	3.06e-02	2.28e-02
S_3^{tot}	5**e**-02	1.25e-01	5.13e-02	1.31e+00	4.92e-02
S_4^{tot}	3**e**-01	1.51e-01	8.23e-03	3.84e-01	1.93e-02
S_5^{tot}	2**e**-04	3.45e+02	3.02e+00	8.85e+01	6.78e+00
S_6^{tot}	2**e**-02	1.63e+00	4.87e-02	2.15e+00	7.63e-02
S_{12}	6**e**-03	9.16e-01	1.34e-01	3.21e+00	2.21e-01
S_{14}	5**e**-03	1.91e-01	1.23e-01	8.64e+00	1.31e+00
S_{15}	8**e**-06	8.45e+02	1.72e+02	9.19e+02	9.62e+02
S_{24}	3**e**-03	3.25e-01	1.88e-02	1.37e+01	5.63e-01
S_{45}	1**e**-05	1.41e-01	4.23e-02	4.25e+01	3.87e+01

applications, maximal values, etc. are also calculated, exported and used in various application areas (environmental protection, agriculture, health care, etc.).

Spatial and time discretization makes each of the submodels a huge computational task, challenging for the most powerful supercomputers available nowadays. That is why the parallelization has always been a key point in the computer implementation of DEM since its very early stages. The EMEP grid or some refinement of it is used for spatial discretization of the equations (EMEP is abbreviation of the *European Monitoring and Evaluation Programme*). The condensed CBM-IV (Carbon Bond Mechanism) is used in the chemistry-deposition part. This chemical scheme is one of the most accurate, but computationally expensive as well.

The development and improvements of DEM throughout the years has lead to a variety of different versions with respect to the grid-size/resolution, vertical layering (2D or 3D model respectively) and the number of species in the chemical scheme. The most prospective of them have been united in the package UNI-DEM. A coarse-grain parallelization strategy based on partitioning of the spatial domain in strips or blocks is mainly used in UNI-DEM. For the purpose of sensitivity analysis studies a specialized version of this package, called SA-DEM, was recently developed. More details about its parallel implementation properties can be found in [25–27]. The use of its new modification for the purpose of our current study is given in the next section.

2 Description of the Danish Eulerian Model and UNI-DEM

The Danish Eulerian Model (DEM) is a complex and powerful large-scale air pollution for calculating the concentrations of a large number of pollutants and other chemical species in the air and their variation along certain time period (month, year, decade, etc.). The concentrations (and other application-oriented functions dependent on them) are calculated for a large geographical region (4800 × 4800 km), covering the whole of Europe, the Mediterranean and some parts of Asia and Africa. It takes into account the main physical, chemical and photochemical processes in the atmosphere. Large input data sets are needed for the emissions and the quickly changing meteorological conditions, which must also be taken into account. From the output results one can derive important information for environmental protection, agriculture, health care, etc.

DEM is mathematically represented by the following system of partial differential equations (PDE), in which the unknowns are the concentrations of a large number of chemical species. The main physical and chemical processes (advection, diffusion, chemical reactions, emissions and deposition) are represented by separate terms in the right-hand side of that system.

$$\begin{aligned}\frac{\partial c_s}{\partial t} = &-\frac{\partial (uc_s)}{\partial x} - \frac{\partial (vc_s)}{\partial y} - \frac{\partial (wc_s)}{\partial z} + \\ &+ \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right) + \\ &+ E_s + Q_s(c_1, c_2, \ldots c_q) - (k_{1s} + k_{2s})c_s, \quad s = 1, 2, \ldots q,\end{aligned} \tag{1}$$

where

- c_s – the concentrations of the chemical species;
- $u,\ v,\ w$ – the wind components along the coordinate axes;
- $K_x,\ K_y,\ K_z$ – diffusion coefficients;
- E_s – the emissions;
- $k_{1s},\ k_{2s}$ – dry/wet deposition coefficients;
- $Q_s(c_1, c_2, \ldots c_q)$ – non-linear functions describing the chemical reactions between species under consideration.

The above PDE system is rather complex and not suitable for direct numerical treatment. It is non-linear. The chemical part formally presented by the term $Q_s(c_1, c_2, \ldots c_q)$ leads to a system of stiff ordinary differential equations [46]. Both non-linearity and stiffness are responsible for numerical and computational difficulties arising after some kind of discretization. The difficulties and challenges appear even when advanced numerical schemes are used (see, for instance [46]). For the purpose of its efficient numerical solution, (1) it is split into submodels, which represent the main physical and chemical processes. The sequential splitting [24] is used in the production version of the model, although other splitting methods have also been considered and implemented in some experimental versions [8, 12].

After the splitting process the following 3 submodels are formed:

$$\frac{\partial c_s^{(1)}}{\partial t} = -\frac{\partial (u c_s^{(1)})}{\partial x} - \frac{\partial (v c_s^{(1)})}{\partial y} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s^{(1)}}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s^{(1)}}{\partial y}\right) = A_1 c_s^{(1)}(t)$$

horizontal advection and diffusion

$$\frac{\partial c_s^{(2)}}{\partial t} = E_s + Q_s(c_1^{(2)}, c_2^{(2)}, \ldots c_q^{(2)}) - (k_{1s} + k_{2s}) c_s^{(2)} = A_2 c_s^{(2)}(t)$$

chemistry, emissions and deposition

$$\frac{\partial c_s^{(3)}}{\partial t} = -\frac{\partial (w c_s^{(3)})}{\partial z} + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s^{(3)}}{\partial z}\right) = A_3 c_s^{(3)}(t)$$

vertical transport

The following methods are used in the numerical solution of the submodels:

- *Advection-diffusion part:* Finite elements, followed by predictor-corrector schemes with several different correctors.
- *Chemistry-deposition part:* An improved version of the QSSA (Quasi Steady-State Approximation) [20].
- *Vertical transport:* Finite elements, followed by theta-methods.

Spatial and time discretization makes each of the submodels a huge computational task, challenging for the most powerful supercomputers available nowadays. That is why the parallelization has always been a key point in the computer implementation of DEM since its very early stages. The EMEP grid or some refinement of it is used for spatial discretization of the equations (EMEP is abbreviation of the *European Monitoring and Evaluation Programme*). The condensed CBM-IV (Carbon Bond Mechanism) is used in the chemistry-deposition part. This chemical scheme is one of the most accurate, but computationally expensive as well.

The development and improvements of DEM throughout the years has lead to a variety of different versions with respect to the grid-size/resolution, vertical layering (2D or 3D model respectively) and the number of species in the chemical scheme. The most prospective of them have been united in the package UNI-DEM. A coarse-grain parallelization strategy based on partitioning of the spatial domain in strips or blocks is mainly used in UNI-DEM. For the purpose of sensitivity analysis studies a specialized version of this package, called SA-DEM, was recently developed. More details about its parallel implementation properties can be found in [25–27]. The use of its new modification for the purpose of our current study is given in the next section.

The goal of our particular research of DEM, described in the rest of this paper, is to study the sensitivity of the output (in terms of mean monthly concentrations of several important pollutants) with respect to variation of input emissions of the anthropogenic pollutants. The anthropogenic emissions are determined as vectors with 4 different components $E = (E^N, E^C, E^S, E^A)$, given separately in the input data flow. These components correspond to 4 different groups of pollutants as follows:

- E^N – nitrogen oxides ($NO + NO_2$);
- E^C – anthropogenic hydrocarbons;
- E^S – sulphur dioxide (SO_2);
- E^A – ammonia (NH_3).

The first stage of computations consists of generation of input data necessary for the particular sensitivity analysis study. In our case this means to perform a number of experiments with UNI-DEM by doing certain perturbations in the data for these emissions. The most commonly used output of UNI-DEM are the mean monthly concentrations of a set of dangerous chemical species (or groups of species) in dependence with the particular chemical scheme, calculated in the grid points of the computational domain. With respect to the chemical species, in this paper we focused on the following three:

- ozone (O_3);
- ammonia (NH_3);
- ammonium sulphate and ammonium nitrate ($NH_4SO_4 + NH_4NO_3$).

With respect to the grid points of the computational domain we selected the nearest to the next three European cities with different climate and level of pollution: (i) Milan, (ii) Manchester and (iii) Edinburgh. In principle, the output results for these points of the domain would be most precise if the experiments are done on the highest resolution grid, available in UNI-DEM - (480×480), with a step 10 km. The input emissions data, however, is available only on the coarser (96×96) grid, so it is used after some interpolation process in the finer resolution grids. That is why for this particular study, where sensitivity with respect to the input emissions is evaluated, we chose to use the (96×96) grid-version, which is not so computationally expensive as well.

3 Mathematical Background of the Sensitivity Analysis

Let us consider a scalar function, representing some model output: $u = f(\mathbf{x})$, where the input parameters $\mathbf{x} = (x_1, x_2, \ldots, x_d) \in U^d \equiv [0, 1]^d$ are assumed to be independent (non-correlated) random variables with a known joint probability density function $p(\mathbf{x}) = p(x_1, \ldots, x_d)$. In this way the output u becomes also a random variable (as it is a function of the random vector $\mathbf{x}$) and let $\mathbf{E}$ be its mathematical expectation. Let $\mathbf{D}[\mathbf{E}(u|x_i)]$ be the variance of the conditional expectation of u with respect to x_i and $\mathbf{D}_u$ - the total variance according to u. This indicator is called *first-order sensitivity index* by Sobol [34] or sometimes *correlation ratio*.

3.1 The Total Sensitivity Index (TSI)

Total Sensitivity Index (TSI) [34, 38] of an input parameter $x_i, \quad i \in \{1, \ldots, d\}$ is the sum of the complete set of mutual sensitivity indices of any order (main effect, two-way interactions (second order), three-way interactions (third order), etc.):

$$S_{x_i}^{tot} = S_i + \sum_{l_1 \neq i} S_{il_1} + \sum_{l_1, l_2 \neq i, l_1 < l_2} S_{il_1 l_2} + \ldots + S_{il_1 \ldots l_{d-1}}, \tag{2}$$

where $S_{il_1 \ldots l_{j-1}}$ – jth j-th order sensitivity index for the parameter x_i $(1 \leq j \leq d)$, $j = 1:$ S_i – the "main effect" of x_i. According to the values of their total sensitivity indices, the input parameters can be classified by the level of their importance (very important, important, unimportant, irrelevant). In most practical problems the high dimensional terms are sufficiently small and can be neglected, thus reducing significantly the number of summands in (1). In addition, a numerical approach for evaluating the small sensitivity indices that combines reduction of the mean value and correlated sampling suggested in [38] has been applied. Analyzing the structure of the error in computing the small sensitivity indices one can see that if $\mathbf{D}_{l_1 \ldots l_v} << f_0^2$, then the method is spoilt by loss of accuracy. In a nonlinear large-scale model, like our target one, it happens that even small by value indices may be important enough and must be taken into account. That is why we pay a special attention to the estimation of small sensitivity indices.

3.2 Sobol Approach, Based on HDMR and ANOVA

The Sobol's method is one of the most often used variance-based methods. It is based on a unique decomposition of the model function into orthogonal terms (summands) of increasing dimension and zero means. Its main advantage is computing in a uniform way not only the first order indices, but also the higher order indices (in quite a similar way as the computation of the main effects). The total sensitivity index can then be calculated with just one Monte Carlo integral per factor.

The Sobol method for global sensitivity analysis, applied here, is based on the so-called *High Dimensional Model Representation (HDMR)* (3) of the model function f (integrable) in the d-dimensional factor space:

$$f(\mathbf{x}) = f_0 + \sum_{s=1}^{d} \sum_{l_1 < \ldots < l_s} f_{l_1 \ldots l_s}(x_{l_1}, x_{l_2}, \ldots, x_{l_s}), \tag{3}$$

where f_0 is a constant. The representation (3) is not unique. Sobol has proven that under the conditions (4) for the right-hand-side functions

$$\int_0^1 f_{l_1 \ldots l_s}(x_{l_1}, x_{l_2}, \ldots, x_{l_s})\, \mathrm{d}x_{l_k} = 0, \quad 1 \leq k \leq s, \quad s = 1, \ldots, d \tag{4}$$

the decomposition (3) is unique and is called *ANOVA-HDMR* of the model function $f(\mathbf{x})$ (the abbreviation ANOVA comming from *Analysis of Variances*). Moreover, the functions of the right-hand side can be defined in a unique way by multidimensional integrals as follows (see also [35–37]).

$$f_0 = \int_{U^d} f(\mathbf{x})\mathrm{d}\mathbf{x} \ , \tag{5}$$

$$f_{l_1}(x_{l_1}) = \int_{U^{d-1}} f(\mathbf{x}) \prod_{k \neq l_1} \mathrm{d}\mathbf{x}_k - f_0, \ l_1 \in 1, \ldots, d \ , \tag{6}$$

$$\int_{U^d} f_{i_1 \ldots i_\mu} f_{j_1 \ldots j_\nu} \, \mathrm{d}\mathbf{x} = 0, \quad (i_1, \ldots, i_\mu) \neq (j_1, \ldots, j_\nu), \quad \mu, \nu \in \{1, \ldots, d\} \ . \tag{7}$$

Definition: Sobol global sensitivity indices

$$S_{l_1 \ \ldots \ l_\nu} = \frac{\mathbf{D}_{l_1 \ \ldots \ l_\nu}}{\mathbf{D}}, \quad \nu \in \{1, \ldots, d\} \tag{8}$$

are defined as ratios of the partial variances

$$\mathbf{D}_{l_1 \ \ldots \ l_\nu} = \int f^2_{l_1 \ \ldots \ l_\nu} \mathrm{d}x_{l_1} \ldots \mathrm{d}x_{l_\nu} \tag{9}$$

over the total variance

$$\mathbf{D} = \int_{U^d} f^2(\mathbf{x})\mathrm{d}\mathbf{x} - f_0^2, \qquad \mathbf{D} = \sum_{\nu=1}^{d} \sum_{l_1 < \ldots < l_\nu} \mathbf{D}_{l_1 \ldots l_\nu} \ . \tag{10}$$

From equalities (10) the following properties hold:

$$S_{l_1 \ \ldots \ l_s} \geq 0, \qquad \sum_{s=1}^{d} \sum_{l_1 < \ldots < l_s}^{d} S_{l_1 \ \ldots \ l_s} = 1 \ . \tag{11}$$

4 The Van der Corput Sequence

A quasirandom or low discrepancy sequence is "less random" than a pseudorandom number sequence, but more useful for numerical computation of integrals in higher dimensions, because low discrepancy sequences tend to sample space "more uniformly" than random numbers.

Let $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(s)})$ for $i = 1, 2, \ldots$. The star discrepancy is given by [22]:

$$D_N^* = D_N^*(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sup_{\Omega \subset E^s} \left| \frac{\#\{\mathbf{x}_n \in \Omega\}}{N} - V(\Omega) \right|, \tag{12}$$

where $E^s = [0, 1)^s$. Let $n = \ldots a_3(n), a_2(n), a_1(n)$ be the representation of n in base b. The radical inverse sequence is given by:

$$n = \sum_{i=0}^{\infty} a_{i+1}(n) b^i, \quad \phi_b(n) = \sum_{i=0}^{\infty} a_{i+1}(n) b^{-(i+1)} \tag{13}$$

and it is fulfilled that

$$D_N^* = O\left(\frac{log N}{N}\right) \tag{14}$$

The ***Van der Corput*** sequence is obtained when $b = 2$ [23, 41].

The van der Corput sequence is often used to generate a "subrandom" sequence of points which have a better covering property than pseudorandom points. The van der Corput sequence generates a sequence of points in [0,1] which never repeats. The elements of the van der Corput sequence are strictly less than 1. The van der Corput sequence writes an integer in a given base b, and then its digits are "reflected" about the decimal point. This maps the numbers from 1 to N into a set of numbers in [0,1], which are especially nicely distributed if N is one less than a power of the base. The generation is quite simple. Given an integer I, the expansion of I in base b is generated. Then, essentially, the result R is generated by writing a decimal point followed by the digits of the expansion of I, in reverse order. This decimal value is actually still in base b, so it must be properly interpreted to generate a usable value. In the numerical experiments we will compare the standard van der Corput sequence with an optimization variant when we increase the base to $b = 2^4$. Such comparison has been made for the first time for the particular model.

The lattice sequence that we are going to use in our experimental study, namely Fibonacci based lattice rule FIBO and polynomial lattice sequence are described in [32, 33, 43].

5 Sensitivity Studies with Respect to Emission Levels

In this section we will present the results for the sensitivity of UNI-DEM output (in particular, the ammonia mean monthly concentrations) with respect to the anthropogenic emissions input data variation are shown and discussed in this section. The anthropogenic emissions input consists of 4 different components

$\mathbf{E^A}$ – ammonia (NH_3);
$\mathbf{E^N}$ – nitrogen oxides ($NO + NO_2$);

$\mathbf{E^S}$ – sulphur dioxide (SO_2);
$\mathbf{E^C}$ – anthropogenic hydrocarbons.

The domain under consideration is the 4-dimensional hypercube $[0.5, 1]^4$). Polynomials of second degree have been used as an approximation tool [11]. The input data have been generated by the improved SA-DEM version, developed for the purpose of our sensitivity studies (see the previous section).

Results of the relative error estimation for the quantities f_0, the total variance **D**, first-order (S_i) and total (S_i^{tot}) sensitivity indices are given in Tables 1, 2, 3, respectively. f_0 is presented by a 4-dimensional integral, while the rest of the above quantities are presented by 8-dimensional integrals, following the ideas of *correlated sampling* technique to compute sensitivity measures in a reliable way (see [21, 39]). The four different stochastic approaches used for numerical integration are presented in separate columns of the tables.

A study of the computational efficiency of the stochastic algorithms under consideration for evaluating sensitivity measures presented by multidimensional integrals (total variance) or a ratio of multidimensional integrals (Sobol global sensitivity indices) have been made. The results show that the computational efficiency of the algorithms depends on integrand dimension and magnitude of estimated quantity. The order of relative error is different for different quantities of interest (see column *Reference value*) for the same sample size.

When $n = 2^{16}$ it can be seen that the modified lattice rule LATM produce better results than the lattice rule based on Fibonacci generalized numbers FIBO - see Table 3. The optimized van der Corput sequence produce worse results than both of the lattice sequences FIBO and LATM. It can be seen that the van der Corput sequence with base $b = 2$ gives worse results by at least 1–2 orders to the optimized van der Corput sequence with base $b = 2^4$ VDCO.

6 Sensitivity Studies with Respect to Chemical Reactions Rates

In this section we will study the sensitivity of the ozone concentration values in the air over Genova with respect to the rate variation of some chemical reactions of the condensed CBM-IV scheme [45], namely: # 1, 3, 7, 22 (time-dependent) and # 27, 28 (time independent). The simplified chemical equations of those reactions are:

$$[\#\mathbf{1}]\ \ NO_2 + h\nu \Longrightarrow NO + O;$$
$$[\#\mathbf{3}]\ \ O_3 + NO \Longrightarrow NO_2;$$
$$[\#\mathbf{7}]\ \ NO_2 + O_3 \Longrightarrow NO_3;$$

[#22] $HO_2 + NO \Longrightarrow OH + NO_2$;
[#27] $HO_2 + HO_2 \Longrightarrow H_2O_2$;
[#28] $OH + CO \Longrightarrow HO_2$.

The domain under consideration is the 6-dimensional hypercube $[0.6, 1.4]^6$). Polynomials of second degree have been used for approximation again (see [10]).

Homma and Saltelli discuss in [21] which is the better estimation of

$$f_0^2 = \left(\int_{U^d} f(\mathbf{x}) \mathrm{d}\mathbf{x} \right)^2 \tag{15}$$

in the expression for total variance and Sobol global sensitivity measures. The first formula is

$$f_0^2 \approx \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,d})\ f(\mathbf{x}'_{i,1}, \ldots, \mathbf{x}'_{i,d}) \tag{16}$$

and the second one is

$$f_0^2 \approx \left\{ \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,d}) \right\}^2 \tag{17}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are two independent sample vectors. In case of estimating sensitivity indices of a fixed order, formula (16) is better (as recommended in [21]), here we use it too.

The relative error estimation for the quantities f_0, the total variance $\mathbf{D}$ and some sensitivity indices are given in Tables 4, 5, 6 respectively. The four different stochastic approaches used for numerical integration are presented in separate columns of the tables. The quantity f_0 is presented by 6-dimensional integral, while the rest are presented by 12-dimensional integrals, following the ideas of *correlated sampling*.

When $n = 2^{16}$ it can be seen that the van der Corput sequences produce similar results to the modified lattice rule LATM and Fibonacci generalized numbers FIBO—see Table 6. When $n = 2^{16}$ the optimized van der Corput sequence VDCO is far better than LATM and FIBO for same of the sensitivity indices - see the value for S_{45} and S_{12} in Table 6. Overall LATM produce better results than the Fibonacci based lattice rule by at least one order—see Table 6. None of the 4 methods estimates S_{15} and S_5 reliably, which has extremely small reference values. This natural "size effect" does not destroy the accuracy of the corresponding total sensitivity indices (which are much larger, so the influence of S_5 and S_{15} is negligible).

From these tables we can see the optimized van der Corput sequence gives better results than both the lattice sequences and the original van der Corput sequence with increasing the dimensionality of the integral. The results obtained by the optimized van der Corput sequence are better for 12 dimensional integrals compared with the results for 8 dimensional integrals. To summarize, Tthe algorithm has been suc-

cessfully applied to compute global Sobol sensitivity measures corresponding to the influence of several input parameters (six chemical reactions rates and four different groups of pollutants) on the concentrations of important air pollutants.

7 Conclusion

The computational efficiency (in terms of relative error and computational time) of several stochastic algorithms for multidimensional numerical integration has been studied to analyze the sensitivity of UNI-DEM model output to variation of input emissions of the anthropogenic pollutants and of rates of several chemical reactions. The algorithms have been successfully applied to compute global Sobol sensitivity measures corresponding to the influence of several input parameters on the concentrations of important air pollutants. The study has been done for the areas of several European cities with different geographical locations. The novelty of the proposed approaches is that the van der Corput sequence and its optimized version have been applied for the first time to sensitivity studies of the particular air pollution model. This is also the first time the Fibonacci based lattice rule and the modified lattice sequence based on transformation function are compared with the van der Corput sequence and its optimized version to the problem under consideration. The numerical tests show that the presented optimized stochastic approach is the most efficient for the multidimensional integrals under consideration and especially for computing small by value sensitivity indices.

The present study focuses on the so-called environmental safety. Environmental security is rapidly becoming a significant topic of present interest all over the world, and environmental modelling has a very high priority in various scientific fields, respectively. Such complicated multidisciplinary problem requires joined research and collaboration between experts in the area of environmental modeling, numerical analysis and scientific computing. The results obtained by our sensitivity analysis, play an extremely important two-sided role: to test and improve mathematical models, and, on the other hand, to reliably interpret the numerical results by relevant specialists. Variance-based analysis is a very effective tool for in-depth study of the relationship between individual parameters, outputs, and internal mechanisms governing the system in question. By identifying the major chemical reactions that affect the behavior of the system, specialists in various fields of application (physics, chemistry) will be able to obtain valuable information about improving the model, which in turn will increase the reliability and sustainability of forecasts. Through a sensitivity analysis, the mathematical model will help to make more accurate estimates of the effects of harmful emissions on human health and losses in agriculture. The progress on the problem of ar pollution modelling is closely related to the development of reliable algorithms for multidimensional numerical integration.

Acknowledgements Venelin Todorov is supported by the by the Bulgarian National Science Fund under Project DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Problems" and by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria. The work is also supported by Young Scientists Project KP-06-M32/2—17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics" and by Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications" funded by National Science Fund—Bulgaria. Barcelona Supercompputing Centre (BSC) is kindly acknowledged too for granting us access and computer time on their most powerful supercomputer.

References

1. Antonov, I., Saleev, V.: An economic method of computing LP_τ-sequences. USSR Comput. Math. Phy. **19**, 252–256 (1979)
2. Bahvalov, N.: On the approximate computation of multiple integrals. Vestn. Mosc. State Univ. **4**, 3–18 (1959)
3. Bakhvalov, N.: On the approximate calculation of multiple integrals. J. Complex. **31**(4), 502–516 (2015)
4. Bratley, P., Fox, B.: Algorithm 659: implementing Sobol's quasirandom sequence generator. ACM Trans. Math. Softw. **14**(1), 88–100 (1988)
5. Cukier, R., Fortuin, C., Shuler, K., Petschek, A., Schaibly, J.: Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I: theory. J. Chem. Phys. **59**, 3873–3878 (1973)
6. Dimitriu, G.: Global sensitivity analysis for a chronic myelogenous leukemia model. In: Proceedings of the 9th International Conference NMA'2018, Borovets, Bulgaria, 20–24 Aug 2018, LNCS 11189. Springer (2019). https://doi.org/10.1007/978-3-030-10692-8_42
7. Dimov, I.T., Atanassov, E.: Exact error estimates and optimal randomized algorithms for integration. LNCS **4310**, 131–139 (2007)
8. Dimov, I.T., Farago, I., Havasi, A., Zlatev, Z.: Operator splitting and commutativity analysis in the Danish Eulerian model. Math. Comp. Sim. **67**, 217–233 (2004)
9. Dimov, I., Georgieva, R.: Monte Carlo algorithms for evaluating Sobol' sensitivity indices. Math. Comput. Simul. **81**(3), 506–514 (2010). ISSN 0378-4754, https://doi.org/10.1016/j.matcom.2009.09.005
10. Dimov, I.T., Georgieva, R., Ostromsky, T., Zlatev, Z.: Variance-based sensitivity analysis of the unified Danish Eulerian model according to variations of chemical rates. In: Dimov, I., Faragó, I., Vulkov, L. (eds.) Proceedings of the NAA 2012, LNCS 8236, pp. 247–254. Springer (2013)
11. Dimov, I.T., Georgieva, R., Ostromsky, T., Zlatev, Z.: Sensitivity studies of pollutant concentrations calculated by UNI-DEM with respect to the input emissions. Cent. Eur. J. Math. **11**(8), 1531–1545 (2013). (Numerical Methods for Large Scale Scientific Computing)
12. Dimov, I.T., Ostromsky, T., Zlatev, Z.: Challenges in using splitting techniques for large-scale environmental modeling. In: Farago, I. (ed.) Advances in Air Pollution Modeling for Environmental Security, NATO Science Series, pp. 115–132. Springer (2005)
13. Doukovska, L.: Constant false alarm rate detectors in intensive noise environment conditions. Cybern. Inf. Technol. **10**(3), 31–48 (2010)
14. Georgiev, I., Kandilarov, J.: An immersed interface FEM for elliptic problems with local own sources. In: AIP Conference Proceedings, vol. 1186, no. 1, pp. 335–342. American Institute of Physics (2009)
15. Georgiev, I.R., Zheleva, I., Filipova, M.: Numerical study of the influence of two-burner heating upon the heat transfer during pyrolysis process used for end-of-life tires (EOLT) treatment. In: AIP Conference Proceedings, vol. 2164

16. Gocheva-Ilieva, Snezhana G., Ivanov, Atanas V., Livieris, Ioannis E.: High performance machine learning models of large scale air pollution data in urban area. Cybern. Inf. Technol. **20**(6), 49–60 (2020)
17. Gocheva-Ilieva, S.G., Ivanov, A.V., Voynikova, D.S., et al.: Time series analysis and forecasting for air pollution in small urban area: an SARIMA and factor analysis approach. Stoch. Environ. Res. Risk Assess. **28**, 1045–1060 (2014)
18. Gocheva-Ilieva, S.G., Voynikova, D.S., Stoimenova, M.P., Ivanov, A.V., Iliev, I.P.: Regression trees modeling of time series for air pollution analysis and forecasting. Neural Comput. Appl. **31**(12), 9023–9039 (2019)
19. Hamdad, H., Pézerat, C., Gauvreau, B., Locqueteau, C., Denoual, Y.: Sensitivity analysis and propagation of uncertainty for the simulation of vehicle pass-by noise. Appl. Acoust. **149**, 85–98 (2019). Elsevier. https://doi.org/10.1016/j.apacoust.2019.01.026
20. Hesstvedt, E., Hov, Ø., Isaksen, I.A.: Quasi-steady-state approximations in air pollution modeling: comparison of two numerical schemes for oxidant prediction. Int. J. Chem. Kinet. **10**, 971–994 (1978)
21. Homma, T., Saltelli, A.: Importance measures in global sensitivity analysis of nonlinear models. Reliab. Eng. Syst. Saf. **52**, 1–17 (1996)
22. Karaivanova, A.: Stochastic numerical methods and simulations (2012)
23. Kuipers, L., Niederreiter, H.: [1974], Uniform distribution of sequences, p. 129,158. Dover Publications (2005). ISBN 0-486-45019-8, Zbl 0281.10001
24. Marchuk, G.I.: Mathematical modeling for the problem of the environment. In: Studies in Mathematics and Applications, North-Holland, Amsterdam, vol. 16 (1985)
25. Ostromsky, T., Dimov, I.T., Georgieva, R., Zlatev, Z.: Air pollution modelling, sensitivity analysis and parallel implementation. Int. J. Environ. Pollut. **46**(1/2), 83–96 (2011)
26. Ostromsky, T., Dimov, I.T., Georgieva, R., Zlatev, Z.: Parallel computation of sensitivity analysis data for the Danish Eulerian model, In: Proceedings of the 8th International Conference LSSC'11, 6–10 Oct 2011 Sozopol Bulgaria, LNCS, 7116, pp. 301–309. Springer (2012)
27. Ostromsky, T., Dimov, I.T., Marinov, P., Georgieva, R., Zlatev, Z.: Advanced sensitivity analysis of the Danish Eulerian model in parallel and grid environment. In: Proceedings of the Third International Conference AMiTaNS'11, 20–25 June 2011 Albena Bulgaria, AIP Conference Proceedings, vol. 1404, pp. 225–232 (2011)
28. Poryazov, S.: A suitable unit of sensitivity in telecommunications. TELECOM, 13–14.10.2011. Sofia. ISSN **1314–2690**, 165–172 (2011)
29. Saltelli, A., Chan, K., Scott, M.: Sensitivity Analysis. Probability and Statistics Series, Wiley and Sons publishers (2000)
30. Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., Tarantola, S.: Global Sensitivity Analysis. The Primer, Wiley (2008)
31. Saltelli, A., Tarantola, S., Campolongo, F., Ratto, M.: Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models. Halsted Press, New York (2004)
32. Sloan, I.H., Kachoyan, P.J.: SIAM J. Numer. Anal. **24**, 116–128 (1987)
33. I.H. Sloan and S. Joe, *Lattice methods for multiple Integration*, (Oxford University Press 1994)
34. Sobol', I.M.: Sensitivity estimates for nonlinear mathematical models. Math. Model. Comput. Exp. **1**, 407–414 (1993)
35. Sobol', I.M.: Sensitivity estimates for nonlinear mathematical models. Matem. Model. **2**(1), 112–118 (1990)
36. Sobol', I.M.: Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math. Comput. Simul. **55**(1–3), 271–280 (2001)
37. Sobol', I.M.: Theorem and examples on high dimensional model representation. Reliab. Eng. Syst. Saf. **79**, 187–193 (2003)
38. Sobol, I.M., Myshetskaya, E.: Monte Carlo estimators for small sensitivity indices. Monte Carlo Methods Appl. **13**(5–6), 455–465 (2007)
39. Sobol, I.M., Tarantola, S., Gatelli, D., Kucherenko, S., Mauntz, W.: Estimating the approximation error when fixing unessential factors in global sensitivity analysis. Reliab. Eng. Syst. Saf. **92**, 957–960 (2007)

40. Wang, Y., Hickernell, F.J.: An historical overview of lattice point sets (2002)
41. van der Corput, J.G.: Verteilungsfunktionen (Erste Mitteilung) (PDF). In: Proceedings of the Koninklijke Akademie van Wetenschappen te Amsterdam (in German), vol. 38, pp. 813–821 (1935). Zbl 0012.34705
42. Veleva, E., Georgiev, I.R.: Seasonality of the levels of particulate matter PM10 air pollutant in the city of Ruse, Bulgaria. In: AIP Conference Proceedings, vol. 2302, no. 1, p. 030006. AIP Publishing LLC (2020)
43. Wang, Y., Hickernell, F.J.: An historical overview of lattice point sets.. In: Monte Carlo and Quasi-Monte Carlo Methods 2000, Proceedings of a Conference held at Hong Kong Baptist University, China (2000)
44. Zheleva, I., Georgiev, I., Filipova, M., Menseidov, D.: Mathematical modeling of the heat transfer during pyrolysis process used for end-of-life tires treatment. In: AIP Conference Proceedings, vol. 1895, no. 1, p. 030008. AIP Publishing LLC (2017)
45. Zlatev, Z.: Computer Treatment of Large Air Pollution Models. KLUWER Academic Publishers, Dorsrecht-Boston-London (1995)
46. Zlatev, Z., Dimov, I.T.: Computational and Numerical Challenges in Environmental Modelling. Elsevier, Amsterdam (2006)
47. Zlatev, Z., Dimov, I., Georgiev, K.: Three-dimensional version of the Danish Eulerian model. Zeitschrift für Angewandte Mathematik und Mechanik **76**(S4), 473–476 (1996)

Advanced Stochastic Approaches Based on Lattice Rules for Multiple Integrals in Option Pricing

Venelin Todorov

Abstract In the present paper we present an optimized lattice rule based on a new generating vector. We evaluate European style options with an exponential payoff function with the optimized lattice rule under consideration. The idea is that the value of the option is formulated in terms of the expectation of some random variable; then the average of independent samples of this random variable is used to estimate the value of the option. A short introduction of the theory of lattice rule has been given and the idea of the optimal generating vectors has been presented. We compare the performance of the new stochastic approach with a new optimal generating vector for multiple integrals up to 100 dimensions. The advantages of the optimized lattice rule have been shown.

Keywords Multiple integrals · Optimized lattice rule · Monte Carlo methods

1 Introduction

Nowadays Monte Carlo (MC) and quasi-Monte Carlo (QMC) methods have become a popular computational device for problems in finance [5, 31]. The field of mathematical finance is becoming more sophisticated and quantitative and the scope of its applications is growing [6, 7]. The pricing of options is a very important problem encountered in financial markets today [10, 35]. The famous Black-Scholes model provides explicit closed form solutions for the values of European style call and put options. Options have been widely traded since the creation of an organized exchange in 1973. Much of the focus to-date has been on high-dimensional problems since these are more challenging from a computational viewpoint [17, 27].

V. Todorov (✉)
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria

Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_21

The QMC methods using special deterministic sequences achieve higher accuracy and computational efficiency compared to the MC methods [3] Applications of low discrepancy sequences to financial problems are studied in [6, 34]. Options have been widely traded since the creation of the organized exchange in 1973. The famous Black-Scholes model provides explicit closed form solutions for the values of the European style call and put options. Besides the applications in option pricing, Monte Carlo methods are also used in other problems in finance [11–13, 22, 24] The basic definitions and terminology used in the paper can be found in [11, 13, 19, 31, 42].

Definition 1 A European call (put) option is a contract which gives its holder a right, but not an obligation, to buy (sell) predefined quantity of underlying asset S at a prescribed time T (expiry date) at a prescribed exercise (strike) price E. An American call (put) option is the same as the European call (put), except that the American options can be exercised at any time during their lifetime, not just at expiry.

This feature makes it much harder to model the price of the American call and put options.

For completeness we give here the other definitions as follows: An Asian option is an option whose payoff is dependent on the history of the asset price via an arithmetic or geometric mean of the price over the expired time. A look-back option is an option whose payoff depends not only on the asset price at expiry, but also on the maximum and minimum value of the asset price over a given time prior to expiry.

Definition 2 Risk neutrality is the characteristic ascribed to an investor who is indifferent with respect to the risk. A rigorous definition is based on the order relationship on a space of random variables over an appropriately defined probability space [9, 20]. However, the most important use of this notion is in its application to the risk-neutral evaluation formula. The risk-free interest rate r is an idealized interest rate, which is usually taken to be the interest rate of an appropriate treasury bond.

2 Description of the Option Pricing Problem

The value of an asset S as a function of time is modeled by the stochastic differential equation

$$\mathrm{d}S = \mu S \mathrm{d}t + \sigma S \mathrm{d}X, \tag{1}$$

where σ is the volatility of the asset, characterizing the fluctuations in the price, μ is the drift rate, which is a measure of the average rate of growth of the asset price and $\mathrm{d}X$ is the increment of a standard Wiener process. The well-known Black–Scholes (BS) model for European call options is described with the following partial differential equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0, \tag{2}$$

where r is the risk-free interest rate. The value $V(S,t)$ of the call option satisfies the final condition at the time of expiry $V(S,T) = \max(S-E,0)$, where E is the strike price. The boundary conditions of a Dirichlet type are as follows:

$$V(0,t) = 0, \quad V(S,t) \sim S - Ee^{-r(T-t)}, S \to \infty.$$

The European put options satisfy the same BS equation (2) with final and boundary conditions

$$V(S,T) = \max(E-S,0),$$

$$V(0,t) = Ee^{-r(T-t)}, \quad V(S,t) \sim 0, S \to \infty.$$

The analytical solution of the BS equation for call options is expressed in an explicit closed form as

$$V(S,t) = C(S,t) = S\mathrm{N}(d_1) - Ee^{-r(T-t)}\mathrm{N}(d_2),$$

where

$$d_1 = \frac{\ln(S/E) + (r+\sigma^2/2)(T-t)}{\sigma\sqrt{T-t}},$$

$$d_2 = \frac{\ln(S/E) + (r-\sigma^2/2)(T-t)}{\sigma\sqrt{T-t}},$$

and $\mathrm{N}(z)$ is the cumulative distribution function of the standard normal distribution. Similarly, the European put options price is computed with the formula

$$V(S,t) = P(S,t) = Ee^{-r(T-t)}\mathrm{N}(-d_2) - S\mathrm{N}(-d_1).$$

MC and QMC methods are suitable when the solution is expressed as the expectation of a random variable, which is the case of the risk-neutral evaluation formula for the European options [31]:

$$V(S,t) = E\Big(e^{-r(T-t)}h(S(T)) \mid S(t) = S, \mu = r\Big), \tag{3}$$

where E is the expectation operator and $h(S)$ is the payoff function. The payoff function of the European option satisfies

$$h(S) = \max(S-E,0)$$

for the call option and

$$h(S) = \max(E-S,0)$$

for the put option. We follow the approach of Lai and Spanier [31].

Let's deal with a European call option whose payoff depends on $k > 1$ assets with prices $S_i, i = 1, \ldots, k$. Each asset is modeled by a random walk (1)

$$\mathrm{d}S_i = \mu_i S_i \mathrm{d}t + \sigma_i S_i \mathrm{d}X_i,$$

where σ_i is the annualized standard deviation for the i-th asset and $\mathrm{d}X_i$ is an increment of a Brownian motion. The Brownian motion is a special type Markov stochastic process, which has the property $\mathrm{d}X \sim \mathrm{N}(0, \sqrt{dt})$, where $\mathrm{N}(\mu, \sigma)$ is the normal distribution with mean μ and variance σ^2.

Now suppose that at expiry time T, the payoff is given by $h(S_1', \ldots, S_k')$, where S' denotes the value of the i-th asset at expiry. Assuming risk neutrality, the value of the option satisfies

$$V = e^{-r(T-t)}(2\pi(T-t))^{-k/2}(\det \Sigma)^{-1/2}(\sigma_1 \ldots \sigma_k)^{-1}$$

$$\int_0^\infty \cdots \int_0^\infty \frac{h(S_1', \ldots, S_k')}{S_1' \ldots S_k'}$$

$$\exp\left(-0.5\alpha^\top \Sigma^{-1}\alpha\right) \mathrm{d}S_1' \ldots \mathrm{d}S_k',$$

where

$$\alpha_i = \left(\sigma_i (T-t)^{1/2}\right)^{-1} \left(\ln(S_i'/S_i) - (r - \sigma_i^2/2)(T-t)\right),$$

r is the risk-free interest rate and Σ is the covariance matrix, whose entry with index (i, j) is the covariance $\mathrm{d}X_i$ and $\mathrm{d}X_j$ of the k assets. We will consider the special case when the payoff function is the exponent function which is important for some application in finance, see [31].

3 Efficient Stochastic Approaches

We will make a brief description of the stochastic approaches that we are going to use in our survey. Such comparison, up to 100 dimensions, has been made for the first time for the problem under consideration. Description of the lattice rules can be found in our previous paper [39], where we study integrals up to 30 dimensions, but we optimize and extend this approach by an optimal generating vector, so it is useful to give the definitions and the theory again for the unfamiliar audience.

3.1 The Sobol Sequence

Let $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(s)}),\ i = 1, 2, \ldots$ Then the discrepancy (star discrepancy) of the set is given by the formula [26]:

$$D_N^* = D_N^*(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sup_{\Omega \subset E^s} \left| \frac{\#\{\mathbf{x}_n \in \Omega\}}{N} - V(\Omega) \right|,$$

where $E^s = [0, 1)^s$. The representation of n in base b is presented by the following formula [33]: $n = \ldots a_3(n), a_2(n), a_1(n),\ n > 0, n \in \mathrm{Z}$.

The radical inverse sequence is defined as [32]:

$n = \sum_{i=0}^{\infty} a_{i+1}(n) b^i,\ \ \phi_b(n) = \sum_{i=0}^{\infty} a_{i+1}(n) b^{-(i+1)}$ and its discrepancy satisfies: $D_N^* = O\left(\frac{logN}{N}\right)$. The Van der Corput sequence [40] is obtained when $b = 2$.

Sobol sequence [1, 8, 34] is defined by:

$$\mathbf{x}_k \in \overline{\sigma}_i{}^{(k)}, k = 0, 1, 2, \ldots$$

where $\overline{\sigma}_i{}^{(k)}$, $i \geq 1$ - set of permutations on every $2^k, k = 0, 1, 2, \ldots$ subsequent points of the Van der Corput sequence. In binary we have that:

$x_n^{(k)} = \bigoplus_{i \geq 0} a_{i+1}(n) v_i$, where $v_i,\ i = 1, \ldots, s$ is a set of direction numbers [25].

For the QMC algorithms based on ***Sobol*** sequences we have that the corresponding discrepancy is:

$$D_N^* = O\left(\frac{log^s N}{N}\right).$$

The implementation of the Sobol sequence (SOBOL) that we are using is an adaption of the INSOBL and GOSOBL routines in ACM TOMS Algorithm 647 [21] and ACM TOMS Algorithm 659 [8, 25] and adapts the ideas of Antonov and Saleev [1].

3.2 Adaptive Approach

Adaptive strategy [4, 16, 18] is well known method for evaluation of multidimensional integrals, especially when the integrand function has peculiarities and peaks. Adaptive Monte Carlo methods proposed by Lautrup [14] use a "sequence of refinements" of the original area selected so that the computations to be concentrated in subdomains with computational difficulties. There are various adaptive strategies depending on the technique of adaptation [15]. Our adaptive algorithm (simple adaptive Monte Carlo algorithm) does not use any a priori information about the smoothness of the integrand, but it uses a posteriori information for the variance obtained during calculations. The main idea is a concentration of random points in the subregions where the variance is large (in terms of a preliminary given accuracy), i.e. the approach is based on a recursive partitioning of the integration area using a posteriori error information for the current partition. Let p_j and I_{Ω_j} are the following expressions: $p_j = \int_{\Omega_j} p(\mathbf{x})\, \mathrm{d}\mathbf{x}$ and $I_{\Omega_j} = \int_{\Omega_j} f(\mathbf{x}) p(\mathbf{x})\, \mathrm{d}\mathbf{x}$. Consider now a random point $\xi^{(j)} \in \Omega_j$ with a density function $p(\mathbf{x})/p_j$. In this case

$I_{\Omega_j} = \mathbf{E}\left[\frac{p_j}{N}\sum_{i=1}^{N} f(\xi_i^{(j)})\right] = \mathbf{E}\theta_N$. This adaptive algorithm gives an approximation with an error $\varepsilon \leq c\ N^{-1/2}$, where $c \leq 0.6745\sigma(\theta)$ ($\sigma(\theta)$ is the standard deviation). From the estimation of the error, it can be concluded that, in general, the simple adaptive Monte Carlo algorithm gives an error less than the error of the Plain Monte Carlo algorithm, but the order is the same. The adaptive MC algorithm applied here is described below.

Algorithm

1. **Input data**: *total number of points* $N1$, *constant* $M = 4$*(the initial number of subregions taken), constant* ε *(max value of the variance in each subregion), constant* δ *(maximal admissible number of subregions), d-dimensionality of the initial region/domain, f - the function of interest.*

   1.1. **Calculate** *the number of points to be taken in each subregion* $N = N1/\delta$.

2. **For** $j = 1,\ M^d$:

   2.1. **Calculate** *the approximation of* I_{Ω_j} *and the variance* $\mathbf{D}_{\Omega_j}$ *in subdomain* Ω_j *based on* N *independent realizations of random variable* θ_N;

   2.2. **If** ($\mathbf{D}_{\Omega_j} \geq \varepsilon$) **then**

      2.2.1. **Choose** *the axis direction on which the partition will perform*,

      2.2.2. **Divide** *the current domain into two* $\left(G_{j_1}, G_{j_2}\right)$ *along the chosen direction*,

      2.2.3. **If** *the length of obtained subinterval is less than* δ **then go to** *step 2.2.1* **else** $j = j_1$ G_{j_1} *is the current domain right and* **go to** *step 2.1*;

   2.3. **Else if** ($\mathbf{D}_{\Omega_j} < \varepsilon$) *but an approximation of* $I_{G_{j_2}}$ *has not been calculated yet*, **then** $j = j_2$ G_{j_2} *is the current domain along the corresponding direction right and* **go to** *step 2.1*;

   2.4. **Else if** ($\mathbf{D}_{\Omega_j} < \varepsilon$) *but there are subdomains along the other axis directions*, **then go to** *step 2.1*;

   2.5. **Else** *Accumulation in the approximation* I_N *of* I.

3.3 Lattice Rules

This is the best approach in our study, so we will give the theory in more details.

Lattice point sets are a special type of low-discrepancy stochastic sequences based on the use of deterministic sequences instead of random sequences [41]. The monographs of Sloan and Kachoyan [37], Sloan and Joe [38] and Hua and Wang [23] provide comprehensive expositions on the theory of integration lattices.

In our study we will use the following a particular rank-1 lattice sequence [41]:

$$\mathbf{x}_k = \left\{\frac{k}{N}\mathbf{z}\right\},\ k = 1, \ldots, N, \tag{4}$$

where N is an integer, $N \geq 2$, $\mathbf{z} = (z_1, z_2, \ldots z_s)$ is an integer vector modulo N of dimensionality s called a generator of the set and $\{z\}$ denotes the fractional part of z. We denote by $P_N = \{x_1, x_2, \ldots, x_N\}$, $x_i \in [0, 1)^s$ the integration nodes of the formula.

Definition 3 ([37]) We say that $f(x)$ belongs to the class of functions $E_s^\alpha(c)$ for $\alpha > 1$ and $c > 0$, if f is a periodic function with period 1 for every of its components x_i, $i = 1, 2 \ldots, s$, defined over the unit cube $[0, 1]^s$ and its Fourier coefficients satisfy the following inequalities:

$$|a(m)| < \frac{c}{(\overline{m}_1 \ldots \overline{m}_s)^\alpha}, \tag{5}$$

where

$$\overline{m} = \begin{cases} |m|, & |m| \neq 0, \\ 0, & m = 0, \end{cases}$$

and the constant c does not depend on $m_1, \ldots, m_s$.

Definition 4 For $f \in E_\alpha^s(c)$ the worst case error is defined as [29, 41]

$$P_\alpha(z, N) = \sum_{z.a \equiv 0 \ (mod N), a \neq 0} \frac{c}{(\overline{m}_1 \ldots \overline{m}_s)^\alpha}. \tag{6}$$

The quantity $P_\alpha(N, z)$ and the discrepancy are similar measures of the quality of the lattice point set.

Definition 5 For $f \in E_\alpha^s(c)$ the worst case error is defined as [41]

$$P_\alpha(\mathbf{z}, N) = \sum_{\mathbf{z}.a \equiv 0 \ (mod N), a \neq 0} \frac{c}{(\overline{m}_1 \ldots \overline{m}_s)^\alpha}.$$

The quantity $P_\alpha(N, \mathbf{z})$ and the discrepancy are similar measures of the quality of the lattice point set. In fact the proofs of existence of lattice point sets with low discrepancy and low worst case error are related. Korobov and Bakhvalov (1959) [2, 28] proved that if N is a prime number, then there is an optimal choice of the generating vector $\mathbf{z}$, so that

$$D_N^* = O(N^{-1} \log^s N),$$
$$P_\alpha(\mathbf{z}, N) = \mathcal{O}(N^{-\alpha} \log^{\alpha s} N).$$

In 1959 Bahvalov proved that [2] there exists an optimal choice of the generating vector $\mathbf{z}$, for which the error of integration satisfies

$$\left| \frac{1}{N} \sum_{k=0}^{N-1} f\left(\left\{ \frac{k}{N} \mathbf{z} \right\} \right) - \int_{[0,1)^s} f(u)du \right| \leq cd(s,\alpha) \frac{(\log N)^{\beta(s,\alpha)}}{N^{\alpha}}, \tag{7}$$

for the function $f \in E_s^{\alpha}(c)$, where $\alpha > 1$ and $d(s,\alpha)$, $\beta(s,\alpha)$ do not depend on N. Moreover, if N is a prime number, then $\beta(s,\alpha) = \alpha(s-1)$.

The generating vector $\mathbf{z}$, for which inequality (7) is satisfied, is an optimal generating vector and the point set P_N is a set of good integration points. In this case the numerical integration method is called Good Lattice Point method (GLP) [36]. While the theoretical result establish the existence of optimal generating vectors the difficulty of the construction of GLPs is in the construction of the optimal vectors. The problem is especially difficult with increasing the dimensionality of the integral and requires much computational effort.

Niederreiter (1978) showed that [33]: if N is a composite number, then there exist lattice rules for which:

$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha(s-1)+1}(\frac{N}{\phi(N)})), s \geq 2,$$
$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha}(\frac{N}{\phi(N)} + \frac{\tau(N)}{log(N)})), s = 2,$$
$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha}(s-1)(1 + \frac{\tau(N)}{log^{s-1}(N)})), s \geq 3,$$

where $\phi(N)$ is the Euler totient function and $\tau(N)$ is the number of positive divisors of N. When N is a prime number there is a generating vector z, so that

$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha} \log^{\alpha(s-1)}(N)). \tag{8}$$

To search over all possible s-dimensional lattice points z mod N, or even over all vectors may require much computational effort. Thus, it is desirable to have efficient search schemes for finding the good lattice points. It has been observed that a good lattice point a is one for which the Zaremba figure of merit,

$$\rho = min_{m \in L^{\perp}, m \neq 0}(\overline{m}_1 \ldots \overline{m}_s). \tag{9}$$

is as large as possible. The problem of finding a good lattice point can then be recast in terms of solving diophantine equations [33]. This has led to some explicit constructions of good lattice points.

The first generating vector that we are going to use is based on the generalized Fibonacci numbers of the corresponding dimension. Let $F_n^{(s)}$ is the n-th term of the corresponding generalized Fibonacci sequence [41] of dimensionality s. It's a sum of previous s terms from this sequence:

$$F_n^{(s)} = \sum_{i=n-s}^{n-1} F_i^{(s)}, \quad \text{where } n \text{ is an integer and } n \geq s \tag{10}$$

and the following initial conditions hold:

$$F_0^{(s)} = F_1^{(s)} = \cdots = F_{s-2}^{(s)} = 0, \; F_{s-1}^{(s)} = 1. \tag{11}$$

Consider the following generating vector [41]:

$$\mathbf{z} = (1, F_n^{(s)}(2), \ldots, F_n^{(s)}(s)), \tag{12}$$

where we use that

$$F_n^{(s)}(j) := F_{n+j-1}^{(s)} - \sum_{i=0}^{j-2} F_{n+i}^{(s)} \tag{13}$$

and $F_{n+l}^{(s)}$ ($l = 0, \ldots, j-1$, j is an integer, $2 \leq j \leq s$) is the corresponding term of the generalized Fibonacci sequence of dimensionality s.

The generating vector (12) is transformed into [23, 41]:

$$\mathbf{z} = (1, F_{n-1}^{(s)} + F_{n-2}^{(s)} + \cdots + F_{n-s+1}^{(s)}, \ldots, F_{n-1}^{(s)} + F_{n-2}^{(s)}, F_{n-1}^{(s)}). \tag{14}$$

Hua and Wang in 1981 [23] proved that the lattice point set with $N = F_n^{(s)}$ points obtained by using generating vector based on generalized Fibonacci numbers of corresponding dimensionality

$$\left(\left\{ \frac{1}{F_n^{(s)}} k \right\}, \left\{ \frac{F_n^{(s)}(2)}{F_n^{(s)}} k \right\}, \ldots, \left\{ \frac{F_n^{(s)}(s)}{F_n^{(s)}} k \right\} \right), \tag{15}$$

$1 \leq k \leq F_n^{(s)}$, has discrepancy

$$D^*_{F_n^{(s)}} = \mathcal{O}\left(F_n^{(s)^{-\frac{1}{2} - \frac{1}{2^{s+1} \cdot \ln 2} - \frac{1}{2^{2s+3}}}} \right)$$

and the worst case error is

$$P_\alpha(\mathbf{z}, F_n^{(s)}) = \mathcal{O}\left((F_n^{(s)})^{-\frac{\alpha}{2} - \frac{\alpha}{2^{s+1} \cdot \log 2} - \frac{\alpha}{2^{2s+4}}} \right).$$

We refer to the lattice method with a generating vector Fibonacci numbers as Fibonacci lattice rule (FIBO). The advantages of the FIBO method are the linear complexity of the algorithm and fast computation of the generating vector. The efficiency of the FIBO method for the evaluating European style options is studied in [39].

Table 1 The relative error for 3 dimensional integral

N	OPT	Time	ADAPT	Time	FIBO	Time	SOBOL	Time
19513	1.93e-5	0.01	3.21e-4	2.21	4.69e-4	0.02	4.98e-5	0.56
35890	3.18e-6	0.04	6.55e-5	6.41	5.46e-6	0.06	1.56e-5	1.45
66012	2.65e-6	0.07	5.12e-5	9.86	5.34e-6	0.11	8.11e-6	2.31
121415	9.16e-7	0.12	5.11e-5	15.4	5.34e-6	0.12	3.08e-6	3.80
223317	8.01e-7	0.20	9.34e-5	24.2	1.73e-6	0.22	2.05e-6	6.13

Table 2 The relative error for 3 dimensional integral

Time, s	OPT	ADAPT	FIBO	SOBOL
0.1	9.16e-7	8.67e-4	1.32e-6	3.21e-4
1	6.37e-7	2.96e-5	3.22e-7	8.21e-5
2	4.22e-7	5.45e-4	2.06e-7	2.96e-5
5	1.84e-7	1.14e-4	1.47e-7	5.00e-6
10	6.09e-8	6.56e-5	3.89e-7	2.71e-6
20	1.57e-8	2.04e-5	1.53e-8	1.88e-6

Table 3 The relative error for 5 dimensional integral

N	OPT	Time	ADAPT	Time	FIBO	Time	SOBOL	Time
13624	6.72e-5	0.02	1.89e-3	2.33	9.59e-4	0.03	1.76e-4	0.56
52656	1.53e-5	0.06	2.31e-3	6.18	6.96e-4	0.06	5.05e-5	1.45
103519	8.48e-6	0.09	2.01e-3	9.94	8.72e-5	0.13	2.70e-5	2.52
203513	6.25e-6	0.15	3.42e-4	16.2	8.04e-5	0.25	7.57e-6	6.07
400096	8.16e-7	0.40	9.12e-4	45.6	7.26e-5	0.50	2.52e-6	10.63

If we change the generating vector to be optimal in the way described in [30] we have improved the lattice sequence. The optimal generating vector that we are going to use is constructed recently by Dirk Nuyens [30]. This is a 200-dimensional base-2 generating vector of prime numbers for up to $2^{20} = 1048576$ points. The method is improved by generating the points from a lattice sequence in base 2 in gray coded radical inverse ordering. This generating vector is generated by the fast component-by-component algorithms, developed in his PhD thesis. The special choice of this optimal generating vector is better than the generating vector from generalized Fibonacci numbers for higher dimensions, which is only optimal for the two dimensional case [41].

4 Numerical Examples and Results

The experimental results include the evaluation of the multidimensional integrals

$$I_s = \int\limits_{[0,1]^s} \exp\left(\prod_{i=1}^{s} x_i\right). \tag{16}$$

By expanding the exponential function in Taylor series and integrating the terms $(x_1 \dots x_s)^n$ we obtain

$$\int_{[0,1]^s} \exp\left(\prod_{i=1}^{s} x_i\right) =$$

$$= \sum_{n=0}^{\infty} \frac{1}{(n+1)^s n!} =_s F_s(1, \dots, 1; 2, \dots, 2; 1),$$

where ${}_pF_q(a_1, \dots, a_p; b_1, \dots, b_q; x)$ is the generalized hypergeometric function

$${}_pF_q(a_1, \dots, a_p; b_1, \dots, b_q; x) = \sum_{n=0}^{\infty} \frac{(a_1)_n \dots (a_p)_n}{(b_1)_n \dots (b_q)_n} \frac{x^n}{n!},$$

and $(c)_n = c(c+1)\dots(c+n-1)$ is the Pochhammer symbol.

$$\int\limits_{[0,1]^3} exp(x_1 x_2 x_3) \approx 1.14649907. \tag{17}$$

$$\int\limits_{[0,1]^5} exp(\sum_{i=1}^{5} 0.5 a_i x_i^2 (2 + \sin \sum_{j=1, j\neq i}^{5} x_j)) \approx 2.923651, \tag{18}$$

where $a_i = (1, 0.5, 0.2, 0.2, 0.2)$.

$$\int\limits_{[0,1]^8} exp(\sum_{i=1}^{8} 0.1 x_i) = 1.496805. \tag{19}$$

$$\int\limits_{[0,1]^{20}} exp(\prod_{i=1}^{20} x_i) \approx 1.00000949634. \tag{20}$$

We also compare the performance of the methods on a 50 dimensional integral:

Table 4 Times for 5 dimensional integral

Time, s	OPT	ADAPT	FIBO	SOBOL
0.1	3.07e-6	1.34e-2	7.26e-5	8.22e-4
1	1.32e-6	2.44e-3	2.28e-5	2.91e-4
5	1.13e-6	4.93e-4	5.94e-6	1.71e-5
10	5.47e-7	1.88e-3	3.85e-7	1.79e-5
20	3.52e-7	2.71e-4	7.49e-7	4.96e-6

Table 5 The relative error for 8 dimensional integral

N	OPT	Time	ADAPT	Time	FIBO	Time	SOBOL	Time
16128	1.79e-6	0.04	1.10e-5	12.6	8.08e-4	0.03	8.87e-5	0.13
32192	1.56e-6	0.05	3.32e-5	33.3	1.03e-4	0.07	5.42e-5	0.58
64256	8.01e-7	0.08	4.65e-5	54.2	5.03e-5	0.11	2.34e-5	2.49
128257	6.22e-7	0.13	8.25e-6	88.3	8.13e-6	0.14	4.45e-6	6.36
510994	3.21e-7	0.34	7.07e-6	233.6	5.95e-6	0.57	3.32e-6	19.45

$$I_{50} = \int\limits_{[0,1]^{50}} \exp\left(\prod_{i=1}^{50} x_i\right). \tag{21}$$

We evaluate his reference value by expanding the exponential function in Taylor series and integrating the terms $(x_1 \dots x_{50})^n$ we obtain

$$\int_{[0,1]^{50}} \exp\left(\prod_{i=1}^{50} x_i\right) =$$

$$= \sum_{n=0}^{\infty} \frac{1}{(n+1)^{50} n!} = {}_{50}F_{50}(1, \dots, 1; 2, \dots, 2; 1),$$

where ${}_pF_q(a_1, \dots, a_p; b_1, \dots, b_q; x)$ is the generalized hypergeometric function

$${}_pF_q(a_1, \dots, a_p; b_1, \dots, b_q; x) = \sum_{n=0}^{\infty} \frac{(a_1)_n \dots (a_p)_n}{(b_1)_n \dots (b_q)_n} \frac{x^n}{n!},$$

and $(c)_n = c(c+1)\dots(c+n-1)$ is the Pochhammer symbol.

The experimental results are presented in the Tables. Each table includes the relative error of the QMC method, the CPU-time and the number of integration nodes. The option pricing problem is transformed to evaluation of multidimensional integrals of the exponential function over the unit cube where the values of the

Table 6 Times for the 8 dimensional integral

Time, s	OPT	ADAPT	FIBO	SOBOL
1	2.18e-7	6.34e-4	5.34e-6	2.02e-5
2	1.32e-7	1.58e-4	2.57e-6	2.73e-5
5	9.03e-8	1.44e-4	1.52e-7	8.88e-6
10	5.00e-8	6.61e-5	3.45e-6	5.23e-6
20	2.55e-8	2.77e-5	1.82e-7	2.11e-6

Table 7 The relative error for 20 dimensional integral

N	OPT	Time	ADAPT	Time	FIBO	Time	SOBOL	Time
2048	2.84e-6	0.02	1.14e-2	8.6	8.22e-5	0.03	8.44e-4	0.13
16384	1.04e-6	0.12	4.96e-4	60.3	3.12e-5	0.13	6.82e-5	1.68
65536	9.21e-7	0.91	9.75e-4	474.2	1.36e-5	1.17	8.34e-6	8.69
131072	6.15e-7	2.13	1.25e-5	888.3	8.85e-6	2.34	3.77e-6	14.36
524288	5.33e-8	8.13	1.96e-6	2356	2.15e-6	8.34	1.91e-7	57

Table 8 Times for the 20 dimensional integral

Time, s	OPT	ADAPT	FIBO	SOBOL
1	9.14e-7	1.58e-3	1.48e-5	3.25e-5
2	1.08e-7	1.028e-3	9.17e-6	3.97e-5
5	5.87e-8	8.58e-4	5.19e-6	1.45e-5
10	3.56e-8	4.02e-4	1.73e-6	2.71e-6
20	1.23e-8	1.13e-4	1.38e-7	1.76e-6

Table 9 The relative error for 50 dimensional integral

N	OPT	Time	FIBO	Time	SOBOL	Time
2^{10}	2.88e-7	0.05	6.23e-4	0.08	8.88e-5	3.5
2^{12}	1.88e-7	0.17	1.55e-4	0.35	8.88e-5	16
2^{16}	5.44e-8	2.14	9.72e-5	5.21	8.88e-5	73
2^{20}	4.28e-8	17.65	6.08e-5	32.76	8.88e-6	276

parameters involved in the formula depend the values of the European options. We compare the performance of the optimized lattice rule (OPT), the Fibonacci based lattice rule (FIBO) and two other stochastic approaches - the well known adaptive stochastic approach (ADAPT) [18] and we use the notation SOBOL for the well know Sobol quasi-random sequence [3].

The Adaptive Monte Carlo gives the worst results, because its strength is when the integrand has some pecularities. Sobol is suitable for non-smooth functions and

Table 10 Times for the 50 dimensional integral

Time, s	OPT	FIBO	SOBOL
1	9.14e-8	1.58e-3	1.48e-5
2	3.68e-8	1.028e-3	9.17e-6
5	2.67e-8	8.58e-4	5.19e-6
10	3.34e-9	4.02e-4	1.73e-6
20	1.53e-9	1.13e-4	1.38e-7

it is outperformed by the other methods in low dimensions (Tables 1, 2, 3 and 4). The accuracy of the FIBO method and it has comparable execution time (Table 2). The Fibonacci lattice rule is the fastest method but its accuracy is lower than the accuracy of the lattice rules with increasing the dimensionality of the integral (Tables 5 and 6) and Sobol QMC becomes more accurate than FIBO (Tables 7 and 8). In highest dimension the optimized stochastic approach OPT has the lowest relative error and a very high accuracy regardless of the dimension. (Tables 9 and 10). The advantage of the optimized lattice rule over Fibonacci lattice rule and Sobol approach is even more clear in the 100 dimensional case - see Tables 11 and 12. The main advantage of the presented algorithms is their linear computational complexity with respect to dimension, while the deterministic methods suffer from the so-called curse of dimensionality and they become impractical in higher dimensions. The best performance of the optimized method is explained by the choice of the optimal generating vector, whose computation uses fast component-by-component operations. To summarize the optimized lattice rule gives the best results in terms of lowest relative errors and higher computational efficiency.

5 Conclusion

The applications of optimized lattice rules to the problem of option pricing is studied in the paper. The experimental results from Tables show that the optimized lattice rule with the special choice of the optimal generating vectors has the best performance with respect to relative error and computational time. The progress on the problem of option pricing and the computational finance area is closely related to the development of reliable algorithms for multidimensional numerical integration.

Table 11 Algorithm comparison of the RE for the 100-dimensional integral

N	OPT	Time, s	FIBO	Time, s	SOBOL	Time, s
2^{10}	6.83e-3	0.05	4.13e-1	0.06	6.31e-2	18
2^{12}	3.77e-4	0.17	1.15e-1	0.18	1.23e-2	34
2^{16}	3.36e-5	9.1	6.12e-2	9.2	2.31e-3	170
2^{20}	4.78e-6	57.6	3.18e-2	58.7	2.34e-4	861

Table 12 Algorithm comparison of the RE for the 100-dimensional integral

Time, s	OPT	FIBO	SOBOL
1	2.67e-3	7.18e-2	9.31e-2
2	1.89e-4	6.02e-2	8.66e-2
10	3.22e-5	4.12e-2	6.94e-2
100	8.16e-7	1.13e-2	3.88e-3

Acknowledgements Venelin Todorov is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria and by the Bulgarian National Science Fund under Project KP-06-M32/2 - 17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics". The work is partially supported by the Bulgarian National Science Fund under Project DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Problems".

References

1. Antonov, I., Saleev, V.: An economic method of computing LP_τ-sequences. USSR Comput. Math. Phy. **19**, 252–256 (1979)
2. Bahvalov, N.: On the approximate computation of multiple integrals. Vestnik Moscow State University **4**, 3–18 (1959)
3. Bakhvalov, N.: On the approximate calculation of multiple integrals. J. Complex. **31**(4), 502–516 (2015)
4. Berntsen, J., Espelid, T.O., Genz, A.: An adaptive algorithm for the approximate calculation of multiple integrals. ACM Trans. Math. Softw. **17**, 437–451 (1991)
5. Black, F., Scholes, M.: The pricing of options and corporate liabilities. J. Polit. Econ. **81**(3), 637–659 (1973)
6. Boyle, P.P.: Options: a Monte Carlo approach. J. Financ. Econ. **4**(3), 323–338 (1977)
7. Boyle, P.P., Lai, Y., Tan, K.: Using lattice rules to value low-dimensional derivative contracts (2001)
8. Bratley, P., Fox, B.: Algorithm 659: implementing Sobol's quasirandom sequence generator. ACM Trans. Math. Softw. **14**(1), 88–100 (1988)
9. Broadie, M., Glasserman, P.: Pricing American-style securities using simulation. J. Econ. Dyn. Control **21**(8–9), 1323–1352 (1997)

10. Centeno, V., Georgiev, I.R., Mihova, V., Pavlov, V.: Price forecasting and risk portfolio optimization. In: AIP Conference Proceedings, vol. 2164, no. 1, p. 060006. AIP Publishing LLC (2019)
11. Chance, D..M., Brook, R.: An Introduction to Derivatives and Risk Management, 8th edn. South-Western College Pub (2009)
12. Cox, J.C., Rubinstein, M.: Options Markets, 1st edn. Prentice Hall (1985)
13. Cox, J.C., Ross, S.A., Rubinstein, M.: Option pricing: a simplified approach. J. Financ. Econ. **7**(3), 229–263 (1979)
14. Davis, P.J., Rabinowitz, P.: Methods of Numerical Integration, 2nd edn. Academic Press, London, (1984)
15. Dimov, I.: Monte Carlo Methods for Applied Scientists, p. 291p. World Scientific, New Jersey, London, Singapore (2008)
16. Dimov, I., Georgieva, R.: Monte Carlo algorithms for evaluating Sobol' sensitivity indices. Math. Comput. Simul. **81**(3), 506–514 (2010)
17. Dimov, I., Ivanovska, S., Karaivanova, A.: A Quasi-Monte Carlo Method for Integration with Improved Convergence. Lecture Notes in Computer Science, vol. 2179. Conference of Large-Scale Scientific Computing - LSSC2001 - Sozopol, pp. 158–165. Springer, Berlin, Bulgaria (2001)
18. Dimov, I., Karaivanova, A., Georgieva, R., Ivanovska, S.: Parallel Importance Separation and Adaptive Monte Carlo Algorithms for Multiple Integrals. Springer Lecture Notes in Computer Science, vol. 2542, 99–107 (2003)
19. Duffie, D.: Dynamic Asset Pricing Theory, 3rd edn. Princeton University Press (2001)
20. Duffie, D.: Security Markets: Stochastic Models. Emerald Group Publishing Limited (1988)
21. Fox, B.: Algorithm 647: implementation and relative efficiency of quasirandom sequence generators. ACM Trans. Math. Softw. **12**(4), 362–376 (1986)
22. Gemmill, G.: Options Pricing. McGraw-Hill Publishing Co. (1992)
23. Hua, L.K., Wang, Y.: Applications of Number Theory to Numerical analysis. Springer (1981)
24. Hull, J.C.: Options, Futures, and Other Derivative Securities. 2nd edn. Prentice-Hall (1992)
25. Joe, S., Kuo, F.: Remark on algorithm 659: implementing Sobol's quasirandom sequence generator. ACM Trans. Math. Softw. **29**(1), 49–57 (2003)
26. Kalos, M.A., Whitlock, P.A.: Monte Carlo Methods, Volume 1: Basics. Wiley, New York (1986)
27. Karaivanova, A., Dimov, I.: Error analysis of an adaptive Monte Carlo method for numerical integration. Math. Comput. Simul. **47**, 201–213 (1998)
28. Korobov, N.M.: The approximate computation of multiple integrals. Doklady Akademii Nauk SSSR **124**, 1207–1210 (1959)
29. Korobov, N.M.: Number-Theoretical Methods in Approximate Analysis. Fizmatgiz, Moscow (1963)
30. Kuo, F.Y., Nuyens, D.: Application of quasi-Monte Carlo methods to elliptic PDEs with random diffusion coefficients - a survey of analysis and implementation. Found. Comput. Math. **16**(6), 1631–1696 (2016)
31. Lai, Y., Spanier, J.: Applications of Monte Carlo/Quasi-Monte Carlo methods in finance: option pricing. In: Proceedings of the Claremont Graduate University Conference (1998)
32. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. CBSM 63 (1992)
33. Niederreite, H.: Existence of good lattice points in the sense of Hlawka. Monatsh. Math. **86**, 203–219 (1978)
34. Paskov, S.H.: Computing high dimensional integrals with applications to finance. Technical report CUCS-023-94, Columbia University (1994)
35. Raeva, E., Georgiev, I.R.: Fourier approximation for modeling limit of insurance liability. In: AIP Conference Proceedings, vol. 2025, no. 1, p. 030006. AIP Publishing LLC (2018)
36. Sharygin, I.F.: A lower estimate for the error of quadrature formulas for certain classes of functions. Zh. Vychisl. Mat. i Mat. Fiz. **3**, 370–376 (1963)
37. Sloan, I.H., Kachoyan, P.J.: Lattice methods for multiple integration: theory, error analysis and examples. SIAM J. Numer. Anal. **24**, 116–128 (1987)

38. Sloan, I.H., Joe, S.: Lattice Methods for Multiple Integration. Lattice methods for multiple Integration. Oxford University Press (1994)
39. Todorov, V., Dimov, I., Dimitrov, Y.: Efficient quasi-Monte Carlo methods for multiple integrals in option pricing. In: AIP Conference Proceedings, vol. 2025, no. 1, p. 110007. AIP Publishing LLC (2018)
40. Van der Corput, J.G.: Verteilungsfunktionen (Erste Mitteilung) (PDF). Proceedings of the Koninklijke Akademie van Wetenschappen te Amsterdam (in German) **38**, 813–821 (1935). Zbl 0012.34705
41. Wang, Y., Hickernell, F.J.: An historical overview of lattice point sets. In: Monte Carlo and Quasi-Monte Carlo Methods 2000, Proceedings of a Conference held at Hong Kong Baptist University, China (2000)
42. Wilmott, P., Dewynne, J., Howison, S.: Option Pricing: Mathematical Models and Computation. Oxford University Press (1995)

Advanced Stochastic Approaches for Multidimensional Integrals in Neural Networks

Venelin Todorov, Stefka Fidanova, Ivan Dimov, Stoyan Poryazov, Stoyan Apostolov, and Daniel Todorov

Abstract A very important problem in the area of neural networks and machine learning is the accurate evaluation of multidimensional integrals. An introduction to the theory of the stochastic approaches with a special choice of optimal generating vectors has been given. A new optimized lattice sequence with a special choice of the optimal generating vector have been applied to compute multidimensional integrals up to 100-dimensions. It is shown that the progress in the area of machine learning is strictly related to the progress in reliable algorithms for multidimensional integration.

V. Todorov · S. Poryazov
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
e-mail: stoyan@math.bas.bg

V. Todorov (✉) · S. Fidanova · I. Dimov
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

S. Fidanova
e-mail: stefka@parallel.bas.bg

I. Dimov
e-mail: ivdimov@bas.bg

S. Apostolov
Faculty of Mathematics and Informatics, Sofia University, Sofia, Bulgaria

D. Todorov
Department of Electrical Power Engineering, University of Ruse "Angel Kanchev", Ruse, Bulgaria
e-mail: dltodorov@uni-ruse.bg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_22

1 Introduction

Stochastic techniques have been developed over many years in a range of different fields, but have only recently been applied to the problems in Bayesian statistics [9]. As well as providing a consistent framework for statistical pattern recognition, the stochastic approach offers a number of practical advantages including a solution to the problem for higher dimensions [8]. Multidimensional integrals arise in algebraic analysis for nonidentifiable learning machines [19]. The accurate evaluation of marginal likelihood integrals is a difficult fundamental problem in Bayesian inference that has important applications in machine learning [17]. Conventional approaches to network training are based on the minimization of an error function, and are often motivated by some underlying principle such as maximum likelihood [2]. Such approaches can suffer from a number of deficiencies, including the problem of determining the appropriate level of model complexity. More complex models (e.g. ones with more hidden units or with smaller values of regularization parameters) give better fits to the training data, but if the model is too complex it may give poor generalization (the phenomenon of over-fitting). The usual approach is to set aside data to form a validation set and to optimize the model complexity to give the best validation set performance. The Bayesian viewpoint provides a general and consistent framework for statistical pattern recognition and data analysis. In the context of neural networks, a Bayesian approach offers several important features including the following.

The conventional approach to network training, based on the minimization of an error function, can be seen as a specific approximation to a full Bayesian treatment. Similarly, the technique of regularization arises in a natural way in the Bayesian framework. The corresponding regularization parameters can be treated consistently within the Bayesian setting, without the need for techniques such as cross-validation. For regression problems, error bars, or confidence intervals, can be assigned to the predictions generated by a network. For classification problems, the tendency of conventional approaches to make overconfident predictions in regions of sparse data can be avoided. Bayesian methods provide an objective and principled framework for dealing with the issue of model complexity (for example, how to select the number of hidden units in a feed-forward network), and avoids many of the problems of over-fitting which arise when using maximum likelihood.

2 Problem Settings

Monte Carlo methods are suitable for solving multi-dimensional problems [3, 6, 20], since their computational complexity increases polynomially, but not exponentially with the dimensionality [13, 14]. A general problem in neural networks and machine learning is the accurate evaluation of multidimensional integrals. In 2011 Shaowei Lin in his works [8, 9] consider the problem of evaluating multidimensional integrals

in Bayesian statistics which are used in neuron network and machine learning. We will primarily be interested in two kinds of integrals. The first has the form

$$\int_{\Omega} p_1^{u_1}(x) \dots p_s^{u_s}(x) dx, \tag{1}$$

where $\Omega \in \mathcal{R}^s$, $x = (x_1, \dots, x_s)$, $p_i(x)$ are polynomials and u_i are integers. The second kind of integrals has the form

$$\int_{\Omega} e^{-Nf(x)} \phi(x) dx, \tag{2}$$

where $f(x)$ and $\phi(x)$ are s-dimensional polynomials and N is a natural number.

The asymptotics of such integrals is well understood for regular statistical models, but little was known for singular models until a breakthrough in 2001 due to Sumio Watanabe [19]. His insight was to put the model in a suitable standard form by employing the technique of resolution of singularities from algebraic geometry. To apply the Monte Carlo method we will map the domain Ω into the s dimensional hypercube $[0, 1]^s$. There is always such a transformation, for example $2/\pi \arctan(x)$ maps $(0, \infty)$ to $(0, 1)$.

Consider the problem of approximate integration of the multiple integral

$$\int_{[0,1)^s} f(\mathbf{x}) d\mathbf{x} = \int_0^1 d\mathbf{x}^{(\mathbf{1})} \int_0^1 d\mathbf{x}^{(\mathbf{2})} \dots \int_0^1 d\mathbf{x}^{(\mathbf{s})} f(x^{(1)}, x^{(2)}, \dots, , x^{(s)}) = \theta \tag{3}$$

where $x = (x^{(1)}, \dots, x^{(s)}) \in [0, 1)^s$ and $|\theta| < 1$. Since the variance is finite, the function f must be square-integrable in $[0, 1)^s$.

For small values of s, numerical integration methods such as Simpson's rule or the trapezoidal rule can be used to approximate the integral. These methods, however, suffer from the so-called curse of dimensionality and become impractical as s increases beyond 3 or 4. One viable technique for larger values of s is to use sampling methods such that the estimator of θ becomes $\hat{\theta} = \frac{1}{N} \sum_{n=0}^{N-1} f(x_n)$ where $P_N = \{x_0, x_1, \dots, x_{N-1}\} \in [0, 1)^s$ is some point set. Different techniques are available for selecting these point sets.

When the integration nodes P_N are N independently and identically distributed random points in $[0, 1)^s$, the above sampling method becomes the standard Monte Carlo integration method and the resulting estimator $\hat{\theta}^{MC} \equiv \theta^{MC}$ is known as the Monte Carlo estimator. Important properties of $\hat{\theta}^{MC}$ are as follows:

- It is an unbiased estimator of θ with variance σ^2/N; i.e.

$$\mathbb{E}[\hat{\theta}^{MC}] = \theta \text{and } Var[\hat{\theta}^{MC}] = \sigma^2/N$$

- The Strong Law of Large Numbers asserts that $\hat{\theta}^{MC}$ converges to θ almost surely.

- The Central Limit Theorem guarantees that the distribution of $\hat{\theta}$ converges asymptotically to a normal distribution with mean θ and variance σ^2/N as $N \to \infty$. In other words, the error $|\theta - \hat{\theta}^{MC}|$ converges probabilistically at a rate of $O(N^{-1/2})$.
- The rate of convergence $O(N^{-1/2})$ is independent of s.

2.1 Motivation

Let us consider a simple example of evaluating multidimensional integrals in functional classes which demonstrates the power of the Monte Carlo algorithms [3, 5]. This is a case of practical computations showing high efficiency of the Monte Carlo approach versus the deterministic one. Consider the classical problem of integral evaluation in functional classes with bounded derivatives. Suppose $f(x)$ is a continuous function and let a quadrature formula of Newton or Gauss type be used for calculating the integrals. Consider an example with $d = 100$ (this is a typical number for some applications in control theory and mathematical economics). In order to apply such formulae, we generate a grid in the d-dimensional domain and take the sum (with the respective coefficients according to the chosen formula) of the function values at the grid points. Let a grid be chosen with 15 nodes on the each of the coordinate axes in the d-dimensional cube $G = [0, 1]^d$. In this case we have to compute about 10^{100} values of the function $f(x)$. Suppose a time of $10^{-7}s$ is necessary for calculating one value of the function. Therefore, a time of order 10^{93}s will be necessary for evaluating the integral (remember that 1 year = 31536×10^3s, and that there has been less than 9×10^{10}s since the birth of Pythagoras). If the formula of rectangles (or some similar formula) is applied then the error in the approximate integral calculation is

$$\epsilon \leq cMh^3, \tag{4}$$

where $h = 0.1$ is the mesh size, c is constant independent of h and M is the maximal value of the second derivative [4, 14].

Consider a plain Monte Carlo algorithm for this problem with a probable error of the same order. The algorithm itself consists of generating N pseudo random values (points) (PRV) in G; in calculating the values of $f(x)$ at these points; and averaging the computed values of the function. For each uniformly distributed random point in G we have to generate 100 random numbers uniformly distributed in [0, 1]. To apply the Monte Carlo method it is sufficient that $f(x)$ is continuous. The probable error is:

$$\epsilon \leq 0.6745\sigma(\theta)\frac{1}{\sqrt{N}}, \tag{5}$$

where $\sigma(\theta)$ is the standard deviation of random variable θ for which $E\theta = \int_G f(x)$ $p(x)dx$ and N is the number of realizations of the random variable. The probable error is estimated in [3]:

$$\epsilon \leq 0.6745||f||_{L_2}\frac{1}{\sqrt{N}}. \tag{6}$$

From above equations we conclude that:

$$N \approx (\frac{0.6745||f||_{L_2}}{cM})^2 \times h^{-6}. \tag{7}$$

Suppose that the expression in front of h^{-6} is of order 1. (For many problems it is significantly less than 1 as M is often the maximal value of the second derivative; further the Monte Carlo algorithm can be applied even when it is infinity). For our example $h = 0, 1$, we have $N \approx 10^6$; hence, it will be necessary to generate $100 \times 10^6 = 10 \times 10^7$ PRV. Usually, two operations are sufficient to generate a single PRV. Suppose that the time required to generate one PRV is the same as that for calculating the value of the function at one point in the domain. Therefore, in order to solve the problem with the same accuracy, a time of

$$10 \times 10^7 \times 2 \times 10^{-7} \approx 20s$$

will be necessary. The advantage of the Monte Carlo algorithms to solve such problems is obvious, in the case of 15 dimensional integral it is 3×10^7 times faster than the deterministic one.

3 QMC Methods Based on Lattice Rules

Lattice point sets are a special type of low-discrepancy stochastic sequences based on the use of deterministic sequences instead of random sequences [18]. The monographs of Sloan and Kachoyan [16] and Hua and Wang [7] provide comprehensive expositions on the theory of integration lattices.

In our study we will use the following a particular rank-1 lattice sequence [18]:

$$\mathbf{x}_k = \left\{\frac{k}{N}\mathbf{z}\right\},\ k = 1, \ldots, N, \tag{8}$$

where N is an integer, $N \geq 2$, $\mathbf{z} = (z_1, z_2, \ldots z_s)$ is an integer vector modulo N of dimensionality s called a generator of the set and $\{z\}$ denotes the fractional part of z. We denote by $P_N = \{x_1, x_2, \ldots, x_N\}$, $x_i \in [0, 1)^s$ the integration nodes of the formula.

Definition 1 [16] We say that $f(x)$ belongs to the class of functions $E_s^\alpha(c)$ for $\alpha > 1$ and $c > 0$, if f is a periodic function with period 1 for every of its components $x_i, i = 1, 2 \ldots, s$, defined over the unit cube $[0, 1]^s$ and its Fourier coefficients satisfy the following inequalities:

$$|a(m)| < \frac{c}{(\overline{m}_1 \dots \overline{m}_s)^\alpha}, \tag{9}$$

where

$$\overline{m} = \begin{cases} |m|, & |m| \neq 0, \\ 0, & m = 0, \end{cases}$$

and the constant c does not depend on $m_1, \dots, m_s$.

The discrepancy and the "worst case" error are two important characteristics for the quality of the lattice sequences.

Definition 2 Consider the point set $X = \{x_i \mid i = 1, 2, \dots N\}$ in $[0, 1)^s$ and $N > 1$. Denote by $x_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(s)})$ and $J(v) = [0, v_1) \times [0, v_2) \times \dots \times [0, v_s)$. Then the discrepancy of the set is defined as

$$D(N) := \sup_{0 \le v_j \le 1} \left| \frac{\#\{x_i \in J(v)\}}{N} - \prod_{j=1}^{s} v_j \right|.$$

Definition 3 For $f \in E_\alpha^s(c)$ the worst case error is defined as [18]

$$P_\alpha(z, N) = \sum_{z.a \equiv 0 \ (mod N), a \neq 0} \frac{c}{(\overline{m}_1 \dots \overline{m}_s)^\alpha}.$$

The quantity $P_\alpha(N, z)$ and the discrepancy are similar measures of the quality of the lattice point set.

Definition 4 For $f \in E_\alpha^s(c)$ the worst case error is defined as [18]

$$P_\alpha(\mathbf{z}, N) = \sum_{\mathbf{z}.a \equiv 0 \ (mod N), a \neq 0} \frac{c}{(\overline{m}_1 \dots \overline{m}_s)^\alpha}.$$

The quantity $P_\alpha(N, \mathbf{z})$ and the discrepancy are similar measures of the quality of the lattice point set. In fact the proofs of existence of lattice point sets with low discrepancy and low worst case error are related. Korobov and Bakhvalov (1959) [1][?] proved that if N is a prime number, then there is an optimal choice of the generating vector $\mathbf{z}$, so that

$$D_N^* = O(N^{-1} \log^s N),$$
$$P_\alpha(\mathbf{z}, N) = \mathcal{O}(N^{-\alpha} \log^{\alpha s} N).$$

In 1959 Bahvalov proved that [1] there exists an optimal choice of the generating vector $\mathbf{z}$, for which the error of integration satisfies

$$\left| \frac{1}{N} \sum_{k=0}^{N-1} f\left(\left\{ \frac{k}{N} \mathbf{z} \right\} \right) - \int_{[0,1)^s} f(u)du \right| \leq cd(s,\alpha) \frac{(\log N)^{\beta(s,\alpha)}}{N^{\alpha}}, \tag{10}$$

for the function $f \in E_s^{\alpha}(c)$, where $\alpha > 1$ and $d(s,\alpha)$, $\beta(s,\alpha)$ do not depend on N. Moreover, if N is a prime number, then $\beta(s,\alpha) = \alpha(s-1)$.

The generating vector $\mathbf{z}$, for which inequality (10) is satisfied, is an optimal generating vector and the point set P_N is a set of good integration points. In this case the numerical integration method is called Good Lattice Point method (GLP) [15]. While the theoretical result establish the existence of optimal generating vectors the difficulty of the construction of GLPs is in the construction of the optimal vectors. The problem is especially difficult with increasing the dimensionality of the integral and requires much computational effort.

Niederreiter (1978) showed that [12]:

Theorem 1 *If N is a composite number, then there exist lattice rules for which:*

$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha(s-1)+1}(\frac{N}{\phi(N)})), s \geq 2,$$
$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha}(\frac{N}{\phi(N)} + \frac{\tau(N)}{log(N)})), s = 2,$$
$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha}(\log N)^{\alpha}(s-1)(1 + \frac{\tau(N)}{log^{s-1}(N)})), s \geq 3,$$

where $\phi(N)$ is the Euler totient function and $\tau(N)$ is the number of positive divisors of N. When N is a prime number there is a generating vector z, so that

$$P_\alpha(z,N) = \mathcal{O}(N^{-\alpha} \log^{\alpha(s-1)}(N)).$$

The first generating vector that we are going to use is based on the generalized Fibonacci numbers of the corresponding dimension. Let $F_n^{(s)}$ is the n-th term of the corresponding generalized Fibonacci sequence [18] of dimensionality s. It's a sum of previous s terms from this sequence:

$$F_n^{(s)} = \sum_{i=n-s}^{n-1} F_i^{(s)}, \quad \text{where } n \text{ is an integer and } n \geq s \tag{11}$$

and the following initial conditions hold:

$$F_0^{(s)} = F_1^{(s)} = \ldots = F_{s-2}^{(s)} = 0, \; F_{s-1}^{(s)} = 1. \tag{12}$$

Consider the following generating vector [18]:

$$\mathbf{z} = (1, F_n^{(s)}(2), \ldots, F_n^{(s)}(s)), \tag{13}$$

where we use that

$$F_n^{(s)}(j) := F_{n+j-1}^{(s)} - \sum_{i=0}^{j-2} F_{n+i}^{(s)} \tag{14}$$

and $F_{n+l}^{(s)}$ ($l = 0, \ldots, j-1$, j is an integer, $2 \le j \le s$) is the corresponding term of the generalized Fibonacci sequence of dimensionality s.

The generating vector (13) is transformed into [7, 18]:

$$\mathbf{z} = (1, F_{n-1}^{(s)} + F_{n-2}^{(s)} + \ldots + F_{n-s+1}^{(s)}, \ldots, F_{n-1}^{(s)} + F_{n-2}^{(s)}, F_{n-1}^{(s)}). \tag{15}$$

Hua and Wang in 1981 [7] proved that the lattice point set with $N = F_n^{(s)}$ points obtained by using generating vector based on generalized Fibonacci numbers of corresponding dimensionality

$$\left(\left\{\frac{1}{F_n^{(s)}}k\right\}, \left\{\frac{F_n^{(s)}(2)}{F_n^{(s)}}k\right\}, \ldots, \left\{\frac{F_n^{(s)}(s)}{F_n^{(s)}}k\right\}\right), \tag{16}$$

$1 \le k \le F_n^{(s)}$, has discrepancy

$$D^*_{F_n^{(s)}} = \mathcal{O}\left(F_n^{(s)^{-\frac{1}{2} - \frac{1}{2^{s+1}.\ln 2} - \frac{1}{2^{2s+3}}}}\right)$$

and the worst case error is

$$P_\alpha(\mathbf{z}, F_n^{(s)}) = \mathcal{O}\left((F_n^{(s)})^{-\frac{\alpha}{2} - \frac{\alpha}{2^{s+1}.\log 2} - \frac{\alpha}{2^{2s+4}}}\right).$$

For this Fibonacci based lattice rule we will use the notation FIB.

If we change the generating vector to be optimal in the way described in [10] we have improved the lattice sequence. The optimal generating vector that we are going to use is constructed recently by Dirk Nuyens [10]. This is a 600-dimensional base-2 generating vector of prime numbers for up to $2^{20} = 1048576$ points. The method is improved by generating the points from a lattice sequence in base 2 in gray coded radical inverse ordering. This generating vector is generated by the fast component-by-component algorithms, developed in his PhD thesis. The special choice of this optimal generating vector is better than the generating vector from generalized Fibonacci numbers for higher dimensions, which is only optimal for the two dimensional case [18].

4 Numerical Examples

We considered different examples of 4,7,10,30 and 100 dimensional integrals, respectively, for which we have computed their referent values.
Example 1. s = 4.

$$\int_{[0,1]^4} x_1 x_2^2 e^{x_1 x_2} \sin(x_3)\cos(x_4) \approx 0.108975. \tag{17}$$

Example 2. s = 7.

$$\int_{[0,1]^7} e^{1-\sum_{i=1}^{3} sin(\frac{\pi}{2}.x_i)}.arcsin\left(sin(1)+\frac{\sum_{j=1}^{7} x_j}{200}\right) \approx 0.7515. \tag{18}$$

Example 3. s = 10.

$$\int_{[0,1]^{10}} \frac{4x_1 x_3^2 e^{2x_1 x_3}}{(1+x_2+x_4)^2} e^{x_5+\cdots+x_{10}} \approx 14.808435. \tag{19}$$

Example 4. s = 30.

$$\int_{[0,1]^{30}} \frac{4x_1 x_3^2 e^{2x_1 x_3}}{(1+x_2+x_4)^2} e^{x_5+\cdots+x_{20}} x_{21}\ldots x_{30} \approx 3.244. \tag{20}$$

We also consider the 100-dimensional multidimensional integral defined by the following way:

$$I_{100} = \int_{[0,1]^{100}} \exp\left(\prod_{i=1}^{100} x_i\right), \tag{21}$$

whose reference value is calculated by expanding the exponential function in Taylor series and integrating the terms $(x_1 \cdots x_{100})^n$ namely

$$\int_{[0,1]^{100}} \exp\left(\prod_{i=1}^{100} x_i\right) =$$

$$= \sum_{n=0}^{\infty} \frac{1}{(n+1)^{100} n!} =_{100} F_{100}(1,\cdots,1;2,\cdots,2;1),$$

where ${}_pF_q(a_1, \cdots, a_p; b_1, \cdots, b_q; x)$ is the generalized hypergeometric function

$$ {}_pF_q(a_1, \cdots, a_p; b_1, \cdots, b_q; x) = \sum_{n=0}^{\infty} \frac{(a_1)_n \cdots (a_p)_n}{(b_1)_n \cdots (b_q)_n} \frac{x^n}{n!}, $$

and $(c)_n = c(c+1)\cdots(c+n-1)$ is the Pochhammer symbol.

The results are given in the tables below. We make a comparison between plain Monte Carlo (CRUDE)optimized lattice sequence with an optimal generating vector (OPT), Fibonacci lattice sets (FIBO), Sobol sequence (SOBOL) and Matousek scrambling for Sobol sequence (SCR) [11]. Each Table contains information about the stochastic approach which is applied, the obtained relative error (RE), the needed CPU-time in seconds and the number of points. Note that when the FIBO method is tested, the number of sampled points are always Generalized Fibonacci numbers of the corresponding dimensionality.

Numerical results show essential advantage for the optimized lattice sets algorithm based on an optimal generating vector in comparsion with Fibonacci generalized numbers and Sobol scramble sequence (1–2 orders). For lower dimensions FIBO and Sobol gives results of the same order-see Table 1, 2. For higher dimensions Scramble sequence is better than FIBO and Sobol by at least 1 order. The results for relative errors corresponding to FIBO and Sobol are similar especially for higher sample number, see Tables 3. If the computational time is fixed the advantage of Fibonacci lattice sets in terms of relative error in comparison with Sobol approach is clearly seen, see Tables 4. In general scrambling procedure improves the relative error of the unscrambled nets as it is the case for Sobol sequence and its scrambled version by Matousek linear scrambling as can be seen form the results in Tables 5,6. For very high dimensions the optimized lattice rule outperforms not only the scramble sequence, but also the FIBO method and Sobol sequence by at least 2 orders—see Table 7,8. The experiments show that the optimized lattice sequence with a special choice of the optimal generating vector is the best method in terms of lower relative errors with increasing the dimensionality of the integral—see Table 9,10. The optimized lattice sequence gives the best results compared to the other stochastic approaches with increasing the dimensionality of the multidimensional integral.

Table 1 Algorithm comparison of the RE for the 4-dimensional integral

# of points	OPT	t,s	FIBO	t,s	SOBOL	t,s	SCR	t,s
1490	6.11e-4	0.002	1.01e-3	0.004	9.46e-4	0.43	3.78e-3	0.47
10671	2.13e-5	0.01	8.59e-5	0.02	5.28e-4	1.4	6.10e-4	1.59
20569	6.56e-6	0.02	3.89e-5	0.03	3.52e-5	4.32	1.97e-5	4.54
39648	9.14e-7	0.06	3.01e-5	0.07	2.68e-5	7.77	9.67e-6	8.26
147312	4.78e-7	0.15	3.71e-6	0.24	2.29e-6	23.7	1.40e-6	27.91

Table 2 Algorithm comparison of the rel.errors for the 4-dimensional integral

t, s	OPT	FIBO	SOBOL	SCR
1	5.66e-7	5.62e-6	7.54e-4	6.32e-4
5	3.12e-7	5.38e-7	3.26e-5	1.23e-5
10	5.14e-8	3.77e-7	1.50e-5	8.48e-6
20	3.18e-8	2.67e-8	3.55e-6	1.16e-6

Table 3 Algorithm comparison of the RE for the 7-dimensional integral

# of points	OPT	t,s	FIBO	t,s	SOBOL	t,s	SCR	t,s
2000	6.39e-4	0.14	2.81e-3	0.23	5.45e-3	1.04	2.51e-3	1.42
7936	3.23e-4	0.64	1.38e-3	0.87	1.28e-3	2.08	1.16e-3	3.08
15808	1.23e-5	0.95	9.19e-4	1.73	9.65e-4	3.26	7.58e-4	5.89
62725	3.15e-6	2.54	2.78e-5	3.41	5.18e-4	12.3	3.11e-4	15.64
124946	1.12e-6	6.48	6.87e-5	6.90	1.09e-4	25.4	8.22e-5	31.41

Table 4 Algorithm comparison of the rel.errors for the 7-dimensional integral

t, s	OPT	FIBO	SOBOL	SCR
0.1	7.38e-4	2.38e-3	8.85e-3	8.37e-3
1	1.17e-5	6.19e-4	5.85e-3	1.37e-3
5	2.32e-6	8.81e-5	1.79e-3	8.38e-4
10	9.11e-7	1.88e-5	7.36e-4	4.78e-4
20	7.43e-7	3.87e-6	1.96e-4	9.87e-5

Table 5 Algorithm comparison of the RE for the 10-dimensional integral.

# of points	OPT	t,s	FIBO	t,s	SOBOL	t,s	SCR	t,s
1597	3.14e-4	0.002	4.39e-3	0.003	6.31e-3	0.02	1.46e-3	0.05
17711	6.21e-5	0.02	1.81e-3	0.04	5.31e-4	0.11	1.83e-4	0.21
121393	4.34e-6	0.15	1.20e-3	0.16	1.78e-4	1.21	3.12e-5	1.47
832040	4.11e-7	0.75	1.19e-5	0.70	3.24e-5	12.1	8.25e-6	14.41
3524578	5.32e-8	6.35	2.63e-6	6.45	4.57e-6	121.5	7.71e-7	139.1

Table 6 Algorithm comparison of the rel.errors for the 10-dimensional integral

t, s	OPT	FIBO	SOBOL	SCR
0.1	4.95e-6	9.19e-6	5.31e-4	4.19e-4
1	8.10e-7	5.63e-6	1.81e-4	1.21e-4
5	3.56e-8	2.15e-6	8.07e-5	7.21e-5
10	4.31e-8	1.79e-6	4.77e-5	3.51e-5
20	9.13e-9	8.61e-7	8.42e-6	7.09e-6

Table 7 Algorithm comparison of the RE for the 30-dimensional integral.

# of points	OPT	t,s	SCR	t,s	SOBOL	t,s	FIBO	t,s
1024	1.21e-2	0.02	5.78e-2	0.53	1.18e-1	0.42	8.81e-1	0.02
16384	4.11e-3	0.16	1.53e-2	5.69	8.40e-2	4.5	6.19e-1	0.14
131072	5.24e-4	1.34	1.35e-3	42.1	1.18e-2	30.2	2.78e-1	1.16
1048576	8.81e-5	9.02	6.78e-4	243.9	9.20e-3	168	9.86e-2	8.61

Table 8 Algorithm comparison of the rel.errors for the 30-dimensional integral

Time, sec	OPT	SCR	SOBOL	FIBO
1	3.48e-3	2.38e-2	1.01e-1	2.38e-1
5	4.23e-4	5.46e-3	7.76e-2	1.81e-1
10	8.91e-5	1.25e-3	5.71e-2	9.48e-2
20	2.33e-5	6.11e-4	1.28e-2	7.87e-2

Table 9 The relative error for 100 dimensional integral

N	OPT	Time	FIBO	time	Sobol	time	SCR	time
2^{10}	5.18e-3	0.05	4.13e-1	0.06	8.38e-2	9	6.31e-2	18
2^{12}	3.18e-3	0.17	1.15e-1	0.18	5.68e-2	18	1.23e-2	34
2^{16}	1.44e-4	9.1	6.12e-2	9.2	5.18e-3	93	2.31e-3	170
2^{20}	6.38e-5	57.6	3.18e-2	58.7	1.15e-3	476	2.34e-4	861

Table 10 Times for the 100 dimensional integral

Time,s	OPT	FIBO	Sobol	SCR
1	2.14e-3	7.18e-2	4.58e-1	9.31e-2
2	1.56e-3	6.02e-2	2.57e-1	8.66e-2
10	2.58e-4	4.12e-2	5.45e-2	6.94e-2
100	8.86e-6	1.13e-2	6.58e-3	3.88e-3

5 Conclusion

In this paper an optimized lattice rule has been tested on multidimensional integrals in artificial intelligence. A comprehensive experimental study of optimized lattice rule, Fibonacci lattice sets, Sobol sequence and Matousek scrambling for Sobol sequence has been done on some case test functions. This approaches are the only possible algorithms for high dimensional integrals because the deterministic algorithms need an huge amount of time for the evaluation of the integral. The numerical tests show that the optimized lattice rule is efficient for multidimensional integration and especially for computing high dimensional integrals. It is an important element since this may be crucial in order to achieve a more reliable interpretation of the results in Bayesian statistics which is foundational in artificial intelligence and machine learning.

Acknowledgements The work is supported by Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications" funded by National Science Fund - Bulgaria. Venelin Todorov is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria and by the Bulgarian National Science Fund under Project DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Problems". Stoyan Apostolov is supported by the Bulgarian National Science Fund under Young Scientists Project KP-06-M32/2 - 17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics".

References

1. Bahvalov, N.: On the approximate computation of multiple integrals. Vestnik Moscow State Univ. **4**, 3–18 (1959)
2. Bendavid J., Efficient Monte Carlo Integration Using Boosted Decision Trees and Generative Deep Neural Networks (2017). arXiv:1707.00028
3. Dimov, I.: Monte Carlo Methods for Applied Scientists, p. 291p. World Scientific, New Jersey, London, Singapore (2008)
4. Dimov, I., Karaivanova, A.: Error analysis of an adaptive Monte Carlo method for numerical integration. Math. Comput. Simul. **47**, 201–213 (1998)
5. Dimov, I., Nedjalkov, M., Sellier, J.M.: An Introduction to Applied Quantum Mechanics in the Wigner Monte Carlo Formalism. Phys. Rep. **577**, 1–34 (2015)
6. Georgiev, I. et al.: Comparison of heuristic algorithms for solving a specific model of transportation problem. In: AIP Conference Proceedings. Vol. 2302. No. 1. AIP Publishing LLC (2020)
7. Hua, L.K. and Wang, Y.: Applications of Number Theory to Numerical Analysis (1981)
8. Lin S.: Algebraic Methods for Evaluating Integrals in Bayesian Statistics,'Ph.D. dissertation, UC Berkeley (2011)
9. Lin, S., Sturmfels, B., Xu, Z.: Marginal likelihood integrals for mixtures of independence models. J. Mach. Learn. Res. **10**, 1611–1631 (2009)
10. Kuo, F.Y., Nuyens, D.: Application of quasi-Monte Carlo methods to elliptic PDEs with random diffusion coefficients - a survey of analysis and implementation. Found. Comput. Math. **16**(6), 1631–1696 (2016)
11. Matousek, J.: On the L2-Discrepancy for Anchored Boxes. J. Complex. **14**(4), 527–556 (1998)

12. Niederreiter, H.: Monatsh. Math **86**, 203–219 (1978)
13. Owen A.: Variance and discrepancy with alternative scramblings. ACM Trans. Comput. Logic. 1-16 (2002)
14. Paskov S.: Computing High Dimensional Integrals with Applications to Finance. preprint Columbia University (1994)
15. Sharygin, I.F.: A lower estimate for the error of quadrature formulas for certain classes of functions. Zh. Vychisl. Mat. i Mat. Fiz. **3**, 370–376 (1963)
16. Sloan, I.H., Kachoyan, P.J.: Lattice methods for multiple integration: Theory, error analysis and examples. SIAM J. Numer. Anal. **24**, 116–128 (1987)
17. Song, J., Zhao, S., Ermon, S.: A-nice-mc: Adversarial training for mcmc. In: Advances in Neural Information Processing Systems, pp. 5140–5150 (2017)
18. Wang Y., Hickernell F.: An historical overview of lattice point sets (2002)
19. Watanabe, S.: Algebraic analysis for nonidentifiable learning machines. Neural Comput. **13**, 899–933 (2001). April
20. Zheleva, I., Georgiev, I., Filipova, M.: Identification of the influence of the heating upon the heat transfer during pyrolysis process used for End-of-Life tires treatment. In: MATEC Web of Conferences (Vol. 145, p. 03016). EDP Sciences (2018)

Improved Stochastic Approaches for Evaluation of the Wigner Kernel

Venelin Todorov, Ivan Dimov, and Stoyan Poryazov

Abstract We compute the Wigner kernel by means of stochastic approaches. In this paper we study an optimized Adaptive Monte Carlo algorithm for evaluation of the Wigner kernel. This is an important problem in quantum mechanics represented by difficult multidimensional integrals. The goal of our work is to present an improved adaptive Monte Carlo algorithm and to compare the results with other stochastic approaches for computing the Wigner kernel in 3,6,9-dimensional case. It is important that the 12-dimensional case will be considered for the first time. A comprehensive study and an analysis of the computational complexity of the optimized adaptive Monte Carlo algorithm under consideration has also been presented. It can be seen some advantages of the improved adaptive approach over the original one.

1 Introduction

The Monte Carlo (MC) methods are widely used in solving different multidimensional problems by performing realizations of random processes or random variables [3, 6].

One of the best known physicist Richard Feynman formulated the problem of finding an effective and fast algorithm with linear or polynomial computational complexity for computing multidimensional integrals that represent Wigner kernel [7]. More information about the signed particle formulation of a single-body and many-body system can be found in [10–12].

V. Todorov (✉) · S. Poryazov
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
e-mail: venelin@parallel.bas.bg

V. Todorov · I. Dimov
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_23

Up to now the Wigner kernel is calculated with deterministic methods [13, 14, 16] which suffer from the "curse of dimensionality" and that leads to computational times growing exponentially with the problem dimension, while the stochastic MC methods are not affected by the "curse of dimensionality" [3].

The Monte Carlo approach to numerical problems has shown to be remarkably efficient in performing very large computational tasks since it is an embarrassingly parallel technique. Additionally, Monte Carlo methods are well known to keep performance and accuracy with the increase of dimensionality of a given problem, a rather counterintuitive peculiarity not shared by any known deterministic method [10].

Motivated by these very peculiar and desirable computational features, in this work we depict a full Monte Carlo approach to the problem of simulating single- and many-body quantum systems by means of signed particles. In particular we introduce a stochastic technique, based on the adaptive approach, for the computation of the Wigner kernel which, so far, has represented the main bottleneck of this method (it is equivalent to the calculation of a multi-dimensional integral, a problem in which complexity is known to grow exponentially with the dimensions of the problem). The introduction of this stochastic technique for the kernel is twofold: firstly it reduces the complexity of a quantum many-body simulation from non-linear to linear, secondly it introduces an embarrassingly parallel approach to this very demanding problem.

2 Description of the Optimized Adaptive Approach

Adaptive strategy [1, 4, 5, 8] is well known method for evaluation of multidimensional integrals, especially when the integrand function has peculiarities and peaks. Adaptive Monte Carlo methods proposed by Lautrup [2] use a "sequence of refinements" of the original area selected so that the computations to be concentrated in subdomains with computational difficulties. There are various adaptive strategies depending on the technique of adaptation [3]. Our adaptive algorithm (simple adaptive Monte Carlo algorithm) does not use any a priori information about the smoothness of the integrand, but it uses a posteriori information for the variance obtained during calculations. The main idea is a concentration of random points in the subregions where the variance is large (in terms of a preliminary given accuracy), i.e. the approach is based on a recursive partitioning of the integration area using a posteriori error information for the current partition. Let p_j and I_{Ω_j} are the following expressions: $p_j = \int_{\Omega_j} p(\mathbf{x})\,\mathrm{d}\mathbf{x}$ and $I_{\Omega_j} = \int_{\Omega_j} f(\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}$. Consider now a random point $\xi^{(j)} \in \Omega_j$ with a density function $p(\mathbf{x})/p_j$. In this case $I_{\Omega_j} = \mathbf{E}\left[\frac{p_j}{N}\sum_{i=1}^{N} f(\xi_i^{(j)})\right] = \mathbf{E}\theta_N$. This adaptive algorithm gives an approximation with an error $\varepsilon \leq c\,N^{-1/2}$, where $c \leq 0.6745\sigma(\theta)$ ($\sigma(\theta)$ is the standard deviation). From the estimation of the error, it can be concluded that, in general, the simple adaptive Monte Carlo algorithm gives an error less than the error of the Plain Monte Carlo algorithm, but the order is the same.

The optimized adaptive algorithm has higher accuracy than the original Adaptive Monte Carlo algorithm as can be seen from the tables below. The increase of the constant for the initial number of taken subregions $M = 4$ improves the relative error compared with the previous choice $M = 2$ in [15]. The optimized adaptive algorithm is described below.

Algorithm

1. **Input data**: *total number of points* $N1$, *constant* $M = 4$ *(the initial number of subregions taken), constant* ε *(max value of the variance in each subregion), constant* δ *(maximal admissible number of subregions), d-dimensionality of the initial region/domain, f - the function of interest.*

   1.1. **Calculate** *the number of points to be taken in each subregion* $N = N1/\delta$.

2. **For** $j = 1,\ M^d$:

   2.3. **Else if** $(\mathbf{D}_{\Omega_j} < \varepsilon)$ *but an approximation of* $I_{G_{j_2}}$ *has not been calculated yet*, **then** $j = j_2$ G_{j_2} *is the current domain along the corresponding direction right and* **go to** *step 2.1*;

   2.4. **Else if** $(\mathbf{D}_{\Omega_j} < \varepsilon)$ *but there are subdomains along the other axis directions*, **then go to** *step 2.1*;

   2.5. **Else** *Accumulation in the approximation* I_N *of* I.

Computational Complexity

For the simple case when we have the two dimensional case ($N = 2$) and on the first step in the optimized adaptive approach we have $M = 4$ subdomains in our optimized Adaptive approach and

$$\hat{\theta}_N = \frac{1}{N_1}\sum_{i=1}^{N_1}\theta_i + \frac{1}{N_2}\sum_{i=1}^{N_2}\theta_i + \frac{1}{N_3}\sum_{i=1}^{N_3}\theta_i + \frac{1}{N_4}\sum_{i=1}^{N_4}\theta_i,$$

where $N_1 + N_2 + N_3 + N_4 = N$, so we have the same number of operations as the Crude Monte Carlo, which computational complexity is linear [3], to evaluate an approximation of I_{G_j}.

So we choose only $\mathcal{O}(1)$ subdomains where the variance is greater than the parameter ε and this is independent of N. When we divide the domain on every step adaptiveness is not in all subdomains, but only in $\mathcal{O}(1)$ subdomains. At the beginning we have to choose $\frac{N}{k_0}$ random points. After that when dividing the domain into 2^N subdomains, we choose only $\mathcal{O}(1)$ subdomains, this choice is again independent of N. In these subdomains we choose $\frac{N}{k_1}$ points. On the jth step of the Adaptive approach we choose $\mathcal{O}(1)$ subdomains with $\frac{N}{k_j}$ points. We have that $\sum_{j=0}^{i}\frac{1}{k_j} = 1$. Therefore for the computational complexity we obtain

$$\frac{N}{k_0} + \mathcal{O}(1)\frac{N}{k_1} + \cdots + \mathcal{O}(1)\frac{N}{k_i} =$$

$$= N\mathcal{O}(1)\left(\sum_{j=0}^{i}\frac{1}{k_j}\right) = N\mathcal{O}(1) = \mathcal{O}(N).$$

In this way we can conclude that the computational complexity of the optimized Adaptive algorithm is linear.

3 The Presentation of the Wigner Kernel

The necessity of a quantum theory was raised by a series of experimental observations with some extremely small objects such as electrons and other elementary particles, atoms and molecules, which could not be explained with the help of classical mechanics. After the work of several other theoretical physicist, E. Schrodinger finally formulated his equation, summarizing the description of quantum systems in terms of probability amplitudes, or wave-functions. He was awarded a Nobel prize for physics in 1933, mainly because of this result of him. Right after the birth of the Schrodinger equation, other formulations and rules trying to adequately describe quantum mechanics appeared. Here, the work of E. Wigner stands out among these alternative (but completely equivalent) formulations, as an intuitive model which provides a direct connection between classical and quantum physics. Indeed, the theory of Wigner has many similarities with the classical statistical mechanics. The Wigner formalism has been successfully applied to problems related to fields like nuclear physics and comprehension of chemical reactions. Central for the Wigner's theory is the Wigner equation which represented an incredibly challenging mathematical task for years, being a partial integro-differential equation, where the unknown is a function defined over a $2 \times d \times n$ dimensional phase-space, where d is the dimensionality of the space and n is the number of involved particles. The highly multidimensional Wigner kernel is a major component in the Wigner equation. It can be interpreted as the probability for a particle to jump in the phase-space and have a momentum p; this jump happens discontinuously and in discrete amounts equivalent to half the momenta of light quanta, as if the potential were composed of light. A new formulation of quantum mechanics in terms of signed classical field-less particles is presented in [11]. The theory is based on a novel generalized physical interpretation of the mathematical Wigner Monte Carlo method which is able to simulate the time-dependent single- and many-body Wigner equation [10]. The three postulates which completely define the new mathematical formulation of quantum mechanics in terms of signed particles in a single-body context are given below [9, 12] and they are enough to replicate the results of more conventional quantum theories.

Postulate I. Physical systems can be described by means of (virtual) Newtonian particles, i.e. provided with a position $\mathbf{x}$ and a momentum $\mathbf{p}$ simultaneously, which carry a sign which can be positive or negative.

Postulate II. A signed particle, evolving in a potential $V = V(x)$, behaves as a field-less classical point-particle which, during the time interval $\mathrm{d}t$, creates a new pair of signed particles with a probability $\gamma(\mathbf{x}(t))\mathrm{d}t$, where

$$\gamma(\mathbf{x}) = \int_{-\infty}^{+\infty} \mathrm{D}\mathbf{p}' V_W^+(\mathbf{x}; \mathbf{p}') \equiv \lim_{\triangle \mathbf{p}' \to 0^+} \sum_{\mathbf{M}=-\infty}^{+\infty} V_W^+(\mathbf{x}; \mathbf{M} \triangle \mathbf{p}'),$$

where $\hbar = \frac{h}{2\pi}$ is the reduced Planck constant (h) or Dirac constant, $\mathbf{M} = (M_1, M_2, \ldots, M_d)$ is a set of d integers and $V_W^+(\mathbf{x}; \mathbf{p})$ is the positive part of the quantity

$$V_W(\mathbf{x}; \mathbf{p}) = \frac{i}{\pi^d \hbar^{d+1}} \int_{-\infty}^{+\infty} \mathrm{d}\mathbf{x}' e^{-\frac{2i}{\hbar}\mathbf{x}'\mathbf{p}} [V(\mathbf{x} + \mathbf{x}') - V(\mathbf{x} - \mathbf{x}')], \tag{1}$$

known as the Wigner kernel (in a d-dimensional space) [17]. If, at the moment of creation, the parent particle has sign s, position $\mathbf{x}$ and momentum $\mathbf{p}$, the new particles are both located in $\mathbf{x}$, have signs $+s$ and $-s$, and momentum $\mathbf{p} + \mathbf{p}'$ and $\mathbf{p} - \mathbf{p}'$ respectively, with $\mathbf{p}'$ chosen randomly according to the (normalized) probability $\frac{V_W^+(\mathbf{x};\mathbf{p})}{\gamma(\mathbf{x})}$.

Postulate III. Two particles with opposite sign and same phase-space coordinates $(\mathbf{x}, \mathbf{p})$ annihilate.

4 Numerical Examples

The infinite domain of integration can be mapped into the s-dimensional unit hypercube using the following transformation $\frac{1}{2} + \frac{1}{\pi}\arctan(x)$ which maps $(-\infty, \infty)$ to $(0, 1)$. We want to compute (1) in the 3, 6, 9 and for the first time in 12-dimensional case,

$$Vw(x, p) = \int e^{\left(\frac{-i2\sum_{k=1}^{n} x_k' p_k}{\hbar}\right)} \times$$

$$[V(x_1 + x_1', \ldots x_n + x_n') - V(x_1 - x_1', \ldots x_n - x_n')] dx_1' \ldots dx_n',$$

where the Wigner potential is $V = V(x) = \{x_1 \ldots x_n,\ x', x, p, x + x', x - x' \in [0, 1]\}$. It is well known that Wigner kernel has real values [17].

First, we will make a comparison with deterministic method of mid rectangulars, and after that with the well known stochastic approaches of Sobol QMC and Fibonacci based lattice rule FIBO, see [15].

Table 1 Relative errors for the 3, 6 and 9 dimensional integrals

	time (s)	Crude MC	Adaptive MC	Lattice QMC	Sobol QMC
$s = 3$	0.1	6.99e-02	8.73e-04	8.12e-03	1.01e-02
	1	9.56e-03	4.05e-05	5.42e-03	7.27e-03
	10	2.41e-03	9.12e-06	2.11e-03	7.83e-04
	100	1.53e-03	8.18e-07	9.50e-04	2.18e-04
$s = 6$	0.1	6.65e-02	6.81e-04	8.11e-03	2.13e-02
	1	4.54e-02	9.09e-05	9.25e-04	3.31e-03
	10	8.65e-03	8.13e-06	5.11e-04	9.34e-04
	100	6.15e-03	9.08e-07	1.05e-04	1.27e-04
$s = 9$	0.1	7.23e-02	1.73e-03	1.35e-03	5.42e-02
	1	5.61e-02	8.05e-05	8.72e-04	5.59e-03
	10	9.18e-03	6.32e-06	6.51e-04	5.84e-03
	100	7.44e-03	7.58e-07	3.70e-04	6.39e-04

First we will compare the Adaptive MC algorithm with the Fibonacci lattice sets. In Table 1 we report the relative error of the algorithms for preliminary given computational time for numerical integration. Sobol algorithm is better than Crude algorithm, because of the properties of the well distributed sequences. Numerical results show that the Fibonacci lattice algorithm is the fastest and better than Sobol algorithm for smaller computational time, but the both algorithms produce similar results for the same number of samples. This is the first time the Adaptive MC algorithm is applied to this problem. It gives superior results in comparison with the other stochastic algorithms under consideration. The behavior of Wigner kernel explains the advantage and the efficiency of the Adaptive algorithm for this case study.

In Table 2 it can be seen that the optimized stochastic approach gives better results and lower relative errors than the adaptive approach used in our previous study [15]. It can be seen that the computational time for the optimized adaptive MC approach is better than the deterministic method when the dimensionality increases. The advantage of the optimized adaptive algorithm in comparison with the previously used adaptive algorithm is shown on Fig. 1, and the computation of the position of the signs and the peak are given in Fig. 2. The numerical results including relative errors and computational times corresponding to the algorithms under consideration are presented, and the algorithms efficiency is discussed. A numerical comparison for a given number of samples between the adaptive approach (Adapt) used in [15], the Sobol (Sob) and the lattice sequences based on the generalized Fibonacci numbers of the corresponding dimensionality (FIBO) described in [15] and the new optimized Adaptive approach (OptAdapt) has been given in Tables 3, 5, 7. From the all experiments it can be clearly seen that the optimized adaptive approach gives relative errors

Table 2 Relative error of the optimized Adaptive approach, Adaptive approach and the deterministic mid rectangular method

s	N	determ.	t (s)	OptAdapt	t (s)	Adapt	t (s)
3	$32^2 \times 50$	8.51e-03	0.2	1.47e-03	0.1	2.71e-03	0.1
	$32^2 \times 100$	8.21e-03	0.5	1.14e-04	0.21	3.42e-04	0.2
	$64^2 \times 50$	5.76e-03	1	5.12e-0	0.6	7.52e-05	0.55
	$64^2 \times 100$	4.89e-03	1.9	7.11e-06	1.4	1.21e-05	1.3
6	$8^4 \times 50^2$	1.16e-02	41.2	8.64e-05	19.5	9.09e-04	18.1
	$8^4 \times 100^2$	9.75e-03	160.6	5.21e-06	59.4	1.52e-05	57.9
	$16^4 \times 50^2$	7.84e-03	635.2	3.21e-05	321	4.37e-04	311.5
	$16^4 \times 100^2$	2.12e-03	2469.1	2.13e-05	1001.6	3.80e-04	987.1
9	$6^6 \times 16^3$	1.75e-03	835.5	5.11e-05	345	7.62e-05	330.5
	$6^6 \times 32^3$	1.35e-03	5544.1	1.41e-05	2133.6	2.73e-05	2225.1
	$6^6 \times 40^3$	1.12e-03	10684.4	1.67e-06	4531.5	8.12e-06	4491.5

with at least 1 order better than those produced by the adaptive approach for the cost of bigger computational times, because of the increased number of subregions taken in every subdomain M. The adaptive approach itself gives superior results to the other two stochastic approaches as it is completely described in our previous study [15]. The optimized adaptive MC algorithm is the slowest, but it requires smaller number of random points to achieve better accuracy even for higher dimensions and for a fixed computational time it gives the best relative error by at least 1 order, as can be seen from Tables 4, 6, 8. The optimized Adaptive MC approach outperforms the other two approaches FIBO and Sobol QMC by at least $3-5$ even for 12 dimensional case, see Table 9, 10. We should emphasize here that the efficiency of the optimized adaptive MC algorithm under consideration is high when computational peculiarities of the integrand occur only in comparatively small subregion of the initial integration domain. The efficiency of the optimized adaptive MC algorithm is clearly shown in the case of Wigner kernel, where the integrand have computational specialty in the local subarea of the integration domain as it is in the case for the Wigner kernel, see Fig. 1 and how the peak is approximated by the optimized adaptive approach, see Fig. 2. The optimized adaptive MC algorithm under consideration has significantly lower computational complexity than existing deterministic approaches for multidimensional numerical integration and this is demonstrated especially for higher dimensions.

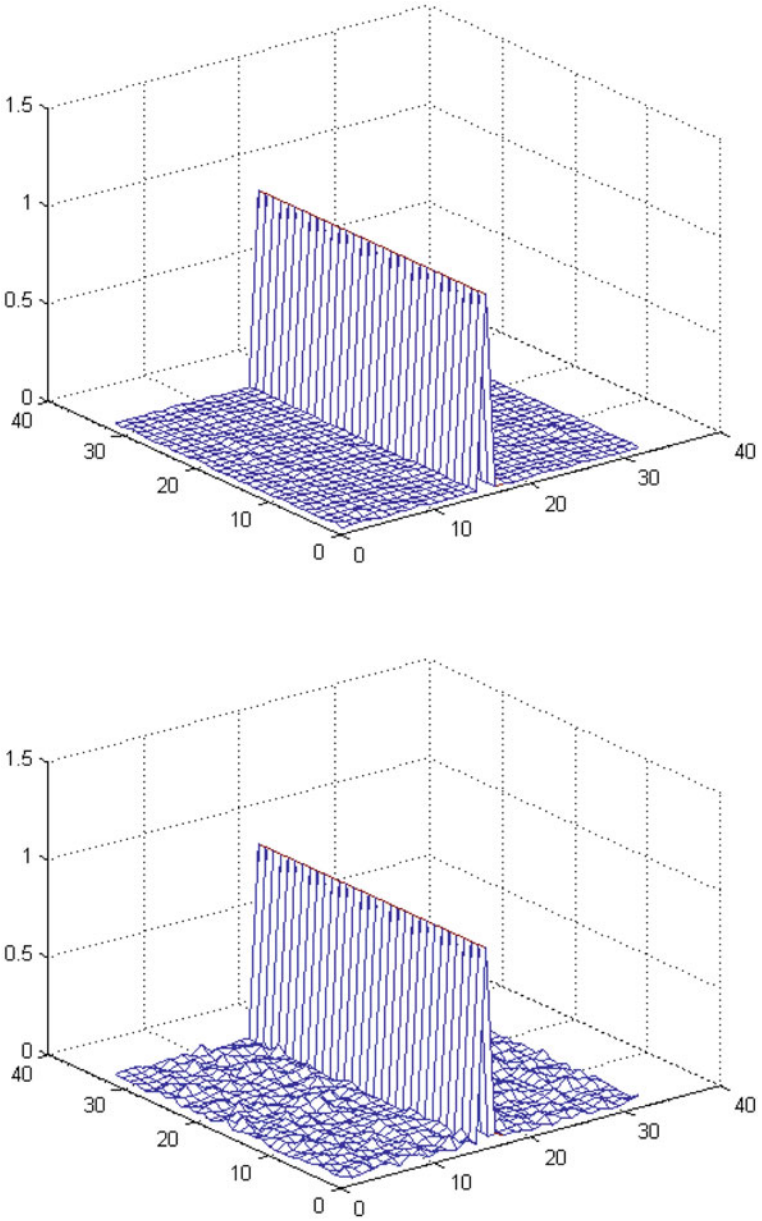


Fig. 1 The Wigner kernel with optimized adaptive and standard adaptive method

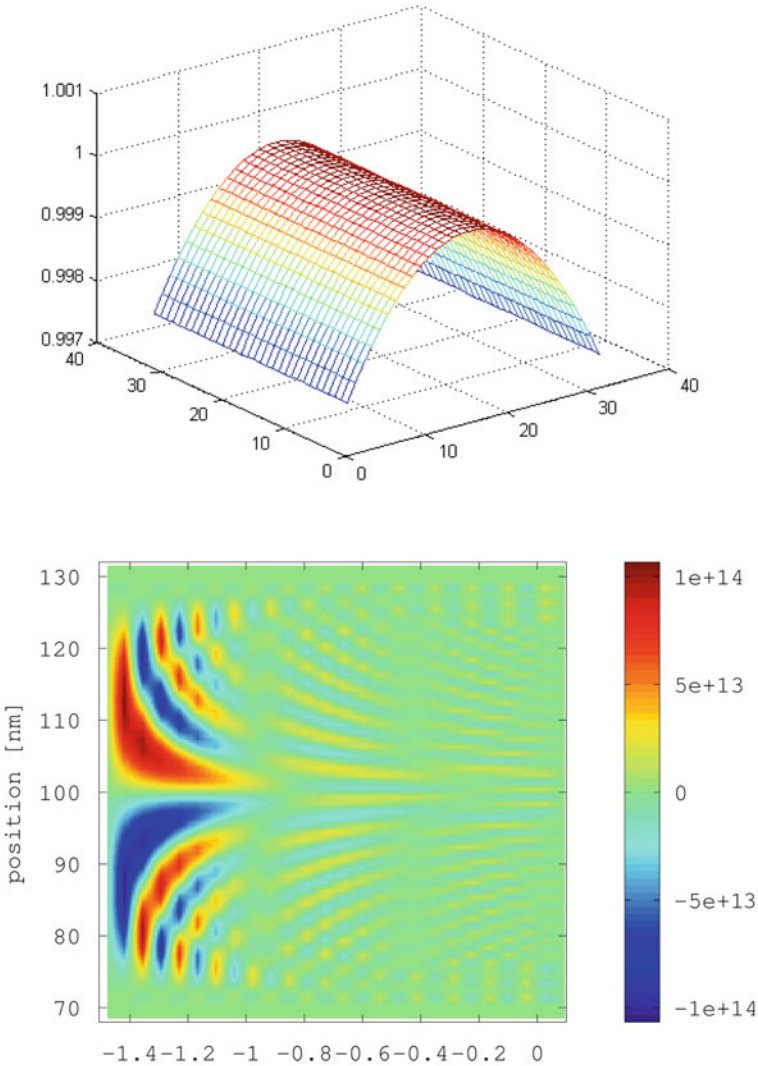


Fig. 2 The position and the peak of the Wigner kernel with optimizes adaptive approach

Table 3 Relative error for 3 dimension

N	Adapt	t,s	OptAdapt	t,s	FIBO	t,s	Sobol	t,s
10^3	5.36e-03	0.3	6.75e-04	0.4	3.72e-02	0.02	1.07e-02	0.05
10^4	4.84e-04	2.9	8.15e-05	3.3	7.06e-03	0.07	8.77e-03	0.54
10^5	2.51e-05	29	5.01e-06	32.6	3.40e-03	0.43	8.57e-04	5.74
10^6	1.76e-05	287	4.38e-07	302	1.01e-03	4.4	6.73e-04	51.6
10^7	6.26e-06	2535	8.02e-08	2708	1.80e-04	49.7	5.98e-05	499

Table 4 Relative error for 3 dimension

time,s	Adapt	OptAdapt	FIBO	Sobol
0.1	6.74e-03	8.73e-04	8.12e-03	1.01e-02
1	8.73e-04	4.05e-05	5.42e-03	7.27e-03
10	5.62e-05	9.12e-06	2.11e-03	7.83e-04
100	3.43e-06	8.18e-07	9.50e-04	2.18e-04

Table 5 Relative error for 6 dimension

N	Adapt	t,s	OptAdapt	t,s	FIBO	t,s	Sobol	t,s
10^3	6.72e-03	0.41	2.23e-04	0.5	7.82e-03	0.01	2.42e-02	0.09
10^4	9.10e-04	3.5	4.74e-05	4.1	5.01e-03	0.07	5.02e-03	0.78
10^5	5.26e-05	33	5.43e-06	37	6.88e-03	0.43	4.60e-04	7.19
10^6	2.70e-06	315	5.04e-07	351	7.68e-04	5.97	3.59e-04	73
10^7	1.03e-06	2438	8.12e-08	2841	4.12e-04	48	8.11e-05	590

Table 6 Relative error for 6 dimension

t,s	Adapt	OptAdapt	FIBO	Sobol
0.1	9.25e-04	6.81e-04	8.11e-03	2.13e-02
1	4.51e-04	9.09e-05	9.25e-04	3.31e-03
10	2.57e-05	8.13e-06	5.11e-04	9.34e-04
100	2.72e-06	5.08e-07	1.05e-04	1.27e-04

Table 7 Relative error for 9 dimension

N	Adapt	t,s	OptAdapt	t,s	FIBO	t,s	Sobol	t,s
10^3	4.92e-02	0.4	8.23e-04	0.5	2.03e-02	0.06	5.42e-02	0.11
10^4	9.09e-04	3.9	2.02e-05	4.7	2.02e-03	0.07	6.02e-03	0.88
10^5	3.32e-05	35	1.08e-06	40	9.16e-04	0.53	3.57e-03	7.56
10^6	6.46e-06	367	4.14e-07	381	7.13e-04	3.7	8.02e-04	72
10^7	1.21e-06	2742	8.91e-08	2912	4.84e-04	40	5.19e-04	621

Table 8 Relative error for 9 dimension

t,s	Adapt	OptAdapt	FIBO	Sobol
0.1	9.24e-03	1.73e-03	1.35e-03	5.42e-02
1	1.23e-03	8.05e-05	8.72e-04	5.59e-03
10	3.82e-05	6.32e-06	6.51e-04	5.84e-03
100	3.09e-06	7.58e-07	3.70e-04	6.39e-04

Table 9 Relative error for 12 dimension

N	Adapt	t,s	OptAdapt	t,s	FIBO	t,s	Sobol	t,s
10^3	3.91e-03	0.7	3.21e-04	0.9	1.33e-02	0.09	2.85e-02	0.2
10^4	5.04e-04	4.5	1.08e-05	6.1	1.34e-03	0.1	4.04e-03	1.34
10^5	2.76e-04	48	5.04e-06	56	5.51e-04	0.68	1.77e-03	9.8
10^6	4.14e-05	415	2.72e-07	432	4.43e-04	4.7	4.07e-04	82
10^7	2.31e-06	3351	4.87e-08	3467	2.5684e-04	60	2.7e-04	700

Table 10 Relative error for 12 dimension

t,s	Adapt	OptAdapt	FIBO	Sobol
0.1	4.66e-03	6.22e-04	6.56e-04	2.67e-02
1	3.25e-04	4.51e-05	4.45e-04	2.98e-03
10	3.24e-05	3.56e-06	3.56e-04	2.45e-03
100	1.31e-05	4.16e-07	1.87e-04	3.21e-04

5 Conclusions

This work shows that the improved adaptive Monte Carlo algorithm under consideration gives the most accurate results in computing the Wigner kernel by a stochastic approach and it has lower computational complexity than the existing deterministic approaches. We should emphasize that the efficiency of the adaptive MC algorithm under consideration is high when computational peculiarities of the integrand occur only in comparatively small subregion of the initial integration domain like in the case of Wigner kernel. This means that the proposed optimized stochastic approach is of great importance for the problems in quantum mechanics with high dimensions. Therefore, the presented improved adaptive MC algorithm is one new successful solution (in terms of robustness and reliability) of Richard Feynman's problem for Wigner kernel evaluation.

Acknowledgements Venelin Todorov is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria and by the Bulgarian National Science Fund under Young Scientists Project KP-06-M32/2 - 17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics". The work is supported by the Bulgarian National Science Fund under Projects DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Problems" and KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications".

References

1. Berntsen, J., Espelid, T.O., Genz, A.: An adaptive algorithm for the approximate calculation of multiple integrals. ACM Trans. Math. Softw. **17**, 437–451 (1991)
2. Davis P.J., Rabinowitz P.: Methods of Numerical Integration, 2nd edn. Academic, London (1984)
3. Dimov I.: Monte Carlo Methods for Applied Scientists, New Jersey, London, Singapore, World Scientific, 291 p., (2008) ISBN-10 981-02-2329-3
4. Dimov, I., Karaivanova, A., Georgieva, R., Ivanovska, S.: Parallel Importance Separation and Adaptive Monte Carlo Algorithms for Multiple Integrals. Springer Lecture Notes in Computer Science, vol. 2542, pp. 99–107 (2003)
5. Dimov, I., Georgieva, R.: Monte Carlo algorithms for evaluating sobol' sensitivity indices. Math. Comput. Simul. **81**(3), 506–514 (2010)
6. Ermakov, S.M.: Monte Carlo Methods and Mixed Problems. Nauka, Moscow (1985)
7. Feynman, R.P.: Space-time approach to non-relativistic quantum mechanics. Rev. Mod Phys. **20** (1948)
8. Georgiev, I., et al.: Comparison of heuristic algorithms for solving a specific model of transportation problem. In: AIP Conference Proceedings, vol. 2302. No. 1. AIP Publishing LLC (2020)
9. Sellier, J.M.: A signed particle formulation of non-relativistic quantum mechanics. J. Comput. Phys. **297**, 254–265 (2015)
10. Sellier, J.M., Dimov, I.: On a full Monte Carlo approach to quantum mechanics. Phys. A **463**, 45–62 (2016)
11. Sellier, J.M., Dimov, I.: The many-body Wigner Monte Carlo method for time-dependent ab-initio quantum simulations. J. Comput. Phys. **273**, 589–597 (2014)
12. Sellier, J.M., Nedjalkov, M., Dimov, I.: An introduction to applied quantum mechanics in the Wigner Monte Carlo formalism. Phys. Rep. **577**, 1–34 (2015)
13. Shao, S., Lu, T., Cai, W.: Adaptive conservative cell average spectral element methods for transient Wigner equation in quantum transport. Commun. Comput. Phys. **9**, 711–739 (2011)
14. Shao, S., Sellier, J.M.: Comparison of deterministic and stochastic methods for time-dependent Wigner simulations. J. Comput. Phys. **300**, 167–185 (2015)
15. Todorov, V., Dimov, I., Georgieva, R., Dimitrov, S.: Adaptive Monte Carlo algorithm for Wigner kernel evaluation. Neural Computing and Applications, 1–12 (2019)
16. Xiong, Y., Chen, Z., Shao, S.: An advective-spectral-mixed method for time-dependent many-body Wigner simulations. SIAM J. Sci. Comput. (2016). arXiv:1602.08853
17. Wigner, E.: On the quantum correction for thermodynamic equilibrium. Phys. Rev. **40**, 749 (1932)

A Numerical Study on Optimal Monte Carlo Algorithm for Multidimensional Integrals

Venelin Todorov, Stoyan Apostolov, Ivan Dimov, Yuri Dimitrov, Stoyan Poryazov, and Daniel Todorov

Abstract In this work we analyze and compare the performance of an optimal stochastic approach for multidimensional integrals of smooth functions. The purpose of the present study is to compare the optimal Monte Carlo algorithm under consideration with the lattice rules based on the generalized Fibonacci numbers of the corresponding dimension and to discuss the advantages and disadvantages of each method. This is the first time this optimal stochastic approach has been compared with other stochastic approaches for mid and high dimensional integrals. The advantages and disadvantages of the stochastic approaches have been discussed.

V. Todorov · Y. Dimitrov · S. Poryazov
Institute of Mathematics and Informatics, Department of Information Modeling, Bulgarian Academy of Sciences, Acad. Georgi Bonchev Str, Block 8, 1113 Sofia, Bulgaria
e-mail: yuri.dimitrov@ltu.bg

S. Poryazov
e-mail: stoyan@math.bas.bg

V. Todorov (✉) · I. Dimov
Institute of Information and Communication Technologies, Department of Parallel Algorithms, Bulgarian Academy of Sciences, Acad. G. Bonchev Str, Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

I. Dimov
e-mail: ivdimov@bas.bg

S. Apostolov
Faculty of Mathematics and Informatics, Sofia University, Sofia, Bulgaria
e-mail: stoyanrapostolov@gmail.com

D. Todorov
Department of Electrical Power Engineering, University of Ruse Angel Kanchev, Ruse, Bulgaria
e-mail: dltodorov@uni-ruse.bg

Y. Dimitrov
Department of Mathematics and Physics, University of Forestry, Sofia, Bulgaria

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_24

1 Introduction

The Monte Carlo (MC) methods are widely used in solving different multidimensional problems by performing realizations of random processes or random variables [14]. The Monte Carlo method has proven to be very useful tool for numerical analysis, particularly when the number of dimension ranging from medium to large [10]. Stochastic approaches based on Monte Carlo methods have many applications [6, 11, 12, 17].

Monte Carlo simulation and quasi-Monte Carlo methods are the prevailing methods used to solve multi-dimensional problems in different areas. Both methods do not suffer from the "curse of the dimensionality" [7].

This is the first time to study optimal stochastic algorithms for computing multidimensional integrals of smooth functions. The optimal approach has been established by Atanasov and Dimov in [2] but there aren't any comparison with other stochastic approaches up to now.

2 Description of the Optimal Monte Carlo Algorithm

Most Monte Carlo methods improve achieves better convergence using the idea of dividing the area of integration [7]. In the case where, in addition to the idea of dividing the area $\Omega = U^s$ (but without the recursive element), the information on the smoothness of the subintegral function is used, an increase in the order of convergence is achieved. Our first known results in this area with probability density $p(\mathbf{x}) = 1$, and splitting the area into equal parts in all directions are obtained by Dupach ([9]).

Theorem 1 *(Dupach, [9]). Let $g(\mathbf{x})$ and all its first-order partial derivatives $\dfrac{\partial g}{\partial x_k}$ are continuous in Ω and bounded, i.e. for all $1 \le k \le s$: $\left|\dfrac{\partial g}{\partial x_k}\right| \le L$, and there exist constants $c_1, c_2 > 0$, for which are fulfilled the following conditions*

$$p_j \le \frac{c_1}{N}, \qquad \text{for } j = 1, \ldots, N, \qquad d_j \le \frac{c_2}{N^{1/s}},$$

where d_j is the diameter of the subdomain Ω_j, i.e. $d_j = \sup\limits_{\mathbf{x}_1, \mathbf{x}_2 \in \Omega_j} |\mathbf{x}_1 - \mathbf{x}_2|$. Then for the estimation of the variance we obtain $\overline{\theta}_N^*$ in the case of $M = N$ and $N_j = 1$, $j = 1, \ldots, M$ and:

$$\mathbf{D}\overline{\theta}_N^* \le c^2 L^2 N^{-1-2/s}, \quad \text{where} \quad c = s c_1 c_2.$$

Using Chebyshev's inequality [20, 21], for the error $\overline{\theta}_N^* - I[g]$ we obtain

$$P\left\{|\overline{\theta}_N^* - I[g]| < \frac{c\,L}{\varepsilon}\,N^{-1/2-1/s}\right\} \geq 1-\varepsilon^2, \tag{1}$$

where ε is a small enough positive number.

Given the Eq. (1) it can be concluded that the probable error $\overline{\theta}_N^* - I[g]$ decreases in order $N^{-1/2-1/s}$, i.e. faster than $N^{-1/2}$. Obviously for large dimensions s the acceleration of the convergence order is relatively insignificant.

The same result for the order of convergence can be achieved in less favorable conditions, namely, only that the corresponding function is continuous. The proof is made by Dimov and Tonev ([8]), and the statement is formulated in the following theorem:

Theorem 2 (Dimov, Tonev; [8]) *Let g is continuous in $\Omega \subset \mathbb{R}^s$ and there exist constants c_1, c_2, c_3, such that $p_j \leq c_1/N,\ d_j \leq c_2 N^{-1/s}$ and $S_j(\cdot, c_3) \subset \Omega_j,\ j = 1, \ldots, N$ (i.e. $M = N$ and $N_j = 1$), where $S_j(\cdot, c_3)$ is sphere with radius c_3. Then for the probable error r_N is fulfilled [7]:*

$$r_N \leq 4\sqrt{2}\frac{c_1 c_2}{c_3}\tau(g; d_{max})_{L_q}\,N^{-1/2-1/s},$$

where $d_{max} = \max_j d_j$ and $\tau(g;\delta)_{L_q}$ is the average modulus of continuity, that is,

$$\tau(g;\delta)_{L_q} = ||\omega(g,\cdot;\delta)||_{L_q} = \left(\int_\Omega (\omega(g,\mathbf{x};\delta))^q\,\mathrm{d}\mathbf{x}\right)^{1/q},$$

$$1 \leq q \leq \infty,\ \delta \in [0, d_{max}] \text{ and}$$

$$\omega(g,\mathbf{x};\delta) = \sup\{|\Delta_h\, g(\mathbf{T})| : \mathbf{T}, \mathbf{T}+h \in [\mathbf{x}-\delta/2, x+\delta/2] \cap \Omega\},$$

where Δ_h is the difference (forward) of the function g in the point $\mathbf{T}$ from first order, which in this case plays a role similar to the derivative but in a discrete sense.

It should be noted that the error of the best quadrature formula according to [16] for the class of functions $W^r_{L_p}(L;[a,b])$, having an absolutely continuous derivative of order $r-1$ and a derivative of order r, which norm in L_p is limited, there is order $\mathcal{O}(N^{-r})$, where N is the number of the nodes of the quadrature formula.

Similar to the one obtained by Dupach, a higher order variance estimate can also be derived for a class of functions whose higher order derivatives are constrained by adding additional conditions for the choice of random points in the subdomains. Rayna Georgieva proves the following theorem [13]:

Theorem 3 *Let we have domain $\Omega \in \mathbb{R}^s$ and the division of the domain $\Omega = \sum_{j=1}^M \Omega_j$, where every subdomain Ω_j is centrally symmetric with center p_j. Let $\xi^{(j)}$ is a random s-dimensional point, uniformly distributed in Ω_j, $\xi^{(j)'}$ is symmetric*

(about the center p_j) with $\xi^{(j)}$, i.e. $\xi^{(j)} + \xi^{(j)'} = 2\ p_j$. If the first and second partial derivatives of a function $g(\mathbf{x})$ are continuous in Ω and for all $k, l = 1, \ldots, s$ is fulfilled that

$$\frac{\partial^2 g}{\partial x_k\ \partial x_l} \leq L \quad \textit{and} \quad d_j = \sup_{x_1, x_2 \in \Omega_j} |x_1 - x_2| \leq \frac{c_1}{N^{1/s}},$$

then

$$\mathbf{D}\,\overline{\theta}_N^{**} = c_2\ L^2\ N^{-1-4/s},$$

where

$$\overline{\theta}_N^{**} = \sum_{j=1}^{N} \frac{V_j}{2}\ [g(\xi^{(j)}) + g(\xi^{(j)'})],$$

V_j is the volume of the subdomain Ω_j and

$$c = (0.5\, s^2 c_1^{s+2})^2.$$

There are various Monte Carlo approaches for numerical integration whose convergence order is $\mathcal{O}\left(N^{-\frac{1}{2}-\frac{k}{s}}\right)$. For $k = 1$ and $k = 2$ these methods can be constructed relatively easily [7], following Dupach's ideas for $k = 1$, described above (see the Theorem 1). But the situation changes when $k \geq 3$. Using the method of controlling the variance on interpolation polynomials, Atanasov and Dimov [2] formulate conditions for constructing a method with optimal order of convergence for s-dimensional functions from the class W^k. The optimal method requires $m * n^s$ points and is briefly described below [2].

For $n, s, k \geq 1$ we define integration formula of type Monte Carlo, depends on an integer parameter $m \geq 1$ and $\binom{s+k-1}{s}$ points in $[0, 1]^s$ by the following way. Points $x^{(r)}$ are exactly a number of $\binom{s+k-1}{s}$ and they must fulfil the condition, that if for one polynom $P(x)$, so that for the degree of the polynom deg $P \leq k$ is fulfilled $P(x^{(r)}) = 0$, then $P \equiv 0$. If $N = n^s$ for $n \geq 1$ we divide s dimensional hypercube $[0, 1]^s$ into n^s *endless undercubes* K_j, i.e. $[0, 1]^s = c_{i=1}^{n^s} K_j$ and

$$K_j = \prod_{i=1}^{s} [a_i^j, b_i^j),$$

where $b_i^j - a_i^j = \frac{1}{n}$ for every $i = 1, \ldots, s$. For every cube K_j we calculate the coordinates of $\binom{s+k-1}{s}$ points $y^{(r)}$, defined by

$$y_i^{(r)} = a_i^r + \frac{1}{n} x_i(r).$$

We assume that we choose m random points

$$\xi_i(j,s) = (\xi_1(j,p), \ldots, \xi_s(j,p))$$

from every cube K_j, such that all the points $\xi_i(j,p)$ are independent uniformly distributed random points, we calculate all $f(y^{(r)})$ and $f(\xi_i(j,p))$, and we calculate the Lagrange polynom of the function f in the point z using information about the functional values in the points $y^{(r)}$. We denote the polynom by $L_k(f,z)$. For every polynom P of max degree $k-1$ we have that $L_k(f,z) \equiv z$. Now we approximate

$$\int_{K_j} f(x)dx \approx \frac{1}{mn^s}\sum_{s=1}^{m}[(\xi(j,p)) - L_k(f,\xi(j,p))] + \int_{K_j} L_k(f,x)dx.$$

After that we sum for every $j = 1, \ldots, N$ and obtain:

$$I(f) \approx \frac{1}{mn^s}\sum_{j=1}^{N}\sum_{s=1}^{m}[(\xi(j,p)) - L_k(f,\xi(j,p))] + \sum_{j=1}^{N}\int_{K_j} L_k(f,x)dx.$$

$$\int_{K_j} f(x)dx \approx \frac{1}{mn^s}\sum_{s=1}^{m}[(\xi(j,p)) - L_k(f,\xi(j,p))] + \int_{K_j} L_k(f,x)dx.$$

After that we sum for every $j = 1, \ldots, N$ and obtain:

$$I(f) \approx \frac{1}{mn^s}\sum_{j=1}^{N}\sum_{s=1}^{m}[(\xi(j,p)) - L_k(f,\xi(j,p))] + \sum_{j=1}^{N}\int_{K_j} L_k(f,x)dx.$$

3 Numerical Examples

We will test the optimal method on multidimensional integrals of smooth functions of different dimensions. We use the following notations: A==Adaptive stochastic approach [3, 7], L=Latin Hypercube sampling [15], S=Sobol quasi-random sequence [1, 4], F=Fibonacci lattice rule [5, 18, 19, 23], O=optimal approach. For more information about the description of the different stochastic approaches, see [20–22]. We must emphasise on the fact this is the first time a comprehensive experimental study with this optimal approach and other stochastic methods under consideration has been done.

Example 1. s = 3

$$\int_{[0,1]^3} \exp(x_1 x_2 x_3) \approx 1.14649907. \tag{2}$$

Example 2. s = 4

$$\int_{[0,1]^4} x_1 x_2^2 e^{x_1 x_2} \sin(x_3) \cos(x_4) \approx 0.1089748630. \tag{3}$$

Example 3. s = 5

$$\int_{[0,1]^5} \exp(-100 x_1 x_2 x_3)(\sin(x_4) + \cos(x_5)) \approx 0.1854297367. \tag{4}$$

Example 4. s= 7.

$$\int_{[0,1]^7} e^{1-\sum_{i=1}^{3} sin(\frac{\pi}{2}.x_i)}.arcsin(sin(1) + \frac{\sum_{j=1}^{7} x_j}{200}) \approx 0.75151101. \tag{5}$$

Example 5. s= 15.

$$\int_{[0,1]^{15}} (\sum_{i=1}^{10} x_i^2)(x_{11} - x_{12}^2 - x_{13}^3 - x_{14}^4 - x_{15}^5)^2 \approx 1.96440666. \tag{6}$$

Example 6. s= 25.

$$\int_{[0,1]^25} \frac{4x_1 x_3^2 e^{2x_1 x_3}}{(1 + x_2 + x_4)^2} e^{x_5 + \cdots + x_{20}} x_{21} \ldots x_{25} \approx 108.808. \tag{7}$$

Example 7. s= 30.

$$\int_{[0,1]^30} \frac{4x_1 x_3^2 e^{2x_1 x_3}}{(1 + x_2 + x_4)^2} e^{x_5 + \cdots + x_{20}} x_{21} \ldots x_{30} \approx 3.244540. \tag{8}$$

For smooth integrands without computational peculiarities Fibonacci lattice algorithm gives better results for lower dimensions, but it requires more random points when the dimensionality increases. The MC algorithm based on Latin hypercube sampling has higher accuracy for this case study with increasing the dimensionality of the integral. The adaptive MC algorithm is slower, but it requires smaller num-

Table 1 Relative error for the 3 dimensional integral dimensional integral

N	A	t	F	t	S	t	L	t	O	t
10^3	4.82e-3	0.17	1.21e-3	0.006	4.87e-4	0.47	6.14e-3	0.004	3.12e-5	0.81
10^4	1.07e-3	1.44	5.04e-4	0.07	1.56e-4	1.88	6.56e-4	0.06	2.05e-6	4.13
10^5	1.52e-4	10.9	5.34e-6	0.66	2.51e-5	15.6	1.34e-4	0.51	4.58e-7	31.62
10^6	5.11e-5	131	7.85e-7	7.02	7.43e-6	105.80	6.84e-5	5.22	6.72e-8	155
10^7	2.34e-5	1094	8.89e-8	79.7	1.58e-6	934	1.73e-5	17	5.34e-9	1053

Table 2 Relative error for the 3 dimensional integral

time(s)	C	A	F	S	L	O
1	1.05e-3	7.96e-3	2.34e-6	2.93e-4	5.11e-4	1.21e-5
5	6.84e-4	8.14e-4	8.47e-7	8.01e-5	7.32e-5	1.12e-6
10	4.79e-4	1.82e-4	4.89e-7	4.71e-5	4.32e-5	7.21e-7
100	1.57e-4	7.04e-5	6.53e-9	7.68e-6	5.32e-6	8.61e-8

Table 3 Relative error for the 4 dimensional integral

N	A	t	F	t	S	t	L	t	O	t
10^4	3.11e-4	1.97	2.61e-5	0.07	2.61e-5	2.14	5.29e-4	0.07	1.52e-5	4.81
10^5	2.44e-5	20.1	5.62e-6	0.99	5.93e-6	17.6	3.56e-4	0.60	7.96e-6	45.1
10^6	1.13e-5	210	5.86e-7	5.22	1.51e-6	193	4.36e-5	4.97	2.31e-7	352.6
10^7	8.11e-6	2035	8.38e-9	58	8.30e-7	1121	8.12e-6	47.1	8.16e-9	2651

Table 4 Relative error for the 4 dimensional integral

time(s)	C	A	F	S	L	O
0.1	3.48e-3	3.59e-4	1.44e-5	4.07e-4	4.18e-4	4.22e-5
1	1.48e-3	2.85e-4	5.62e-6	3.54e-5	3.32e-4	2.31e-5
5	6.62e-4	9.18e-5	5.38e-7	5.26e-5	4.23e-5	1.12e-5
10	2.52e-4	1.36e-5	3.77e-7	6.50e-6	3.48e-5	7.53e-6
20	1.58e-4	2.08e-5	2.67e-8	4.55e-6	2.16e-5	6.54e-7

ber of random points to achieve better accuracy even for higher dimensions. The efficiency of the optimal C algorithm is clearly seen for lower dimensions.

As the dimension increases, the computational time of the optimal method decrease due to the increased computational time of the Lagrangian interpolation polynomial. But for very high dimensions 25 and 30 the relative error for a fixed computational time is better than this achieved by the other stochastic methods. From the Tables it is obvious that for small dimensions the FIBO and Optimal method achieves the best accuracy for a fixed preliminary given time—see Tables 2, 4, 6. For

Table 5 Relative error for the 5 dimensional integral

N	A	t	F	t	S	t	L	t	O	t
10^3	2.15e-3	0.27	1.75e-4	0.007	5.29e-4	0.03	9.38e-3	0.007	2.75e-5	2.1
10^4	2.01e-3	2.43	1.28e-5	0.06	1.43e-4	0.3	3.44e-3	0.07	7.22e-6	2.3
10^5	6.91e-4	22.2	9.50e-6	0.61	2.36e-5	2.77	2.01e-3	0.69	2.36e-6	6.2
10^6	2.92e-4	219.5	5.47e-7	5.98	6.07e-6	24.2	1.80e-4	6.17	5.46e-7	20.0
10^7	8.21e-5	2043	8.71e-9	58.4	2.30e-6	245	2.46e-5	60.5	7.01e-8	105.1

Table 6 Relative error for the 5 dimensional integral

time(s)	A	F	S	L	O
0.1	3.16e-3	3.48e-3	1.09e-5	1.34e-4	3.21e-3
1	1.08e-3	2.08e-3	5.58e-6	7.21e-5	8.54e-4
5	8.79e-4	8.20e-4	8.71e-7	1.54e-5	3.25e-4
10	5.85e-4	7.51e-4	4.15e-7	9.32e-6	8.65e-5
20	3.99e-4	6.95e-4	8.37e-8	7.39e-6	5.02e-5

Table 7 Relative error for the 7 dimensional integral

N	A	t	F	t	S	t	L	t	O	t
10^4	1.07e-3	2.07	2.19e-3	0.11	2.27e-4	0.76	1.79e-3	0.13	2.13e-4	10.2
10^5	7.51e-4	19.3	6.19e-4	0.99	1.22e-4	7.45	2.53e-4	1.15	4.41e-5	40.2
10^6	6.30e-5	194	1.99e-5	9.81	4.71e-5	72.3	8.27e-5	10.32	1.27e-6	167.1
10^7	2.34e-5	1861	4.89e-7	94.2	9.45e-6	697	1.69e-5	101.2	1.45e-7	595.1

Table 8 Relative error for the 7 dimensional integral

time(s)	A	F	S	L	O
0.1	2.38e-2	3.11e-2	2.38e-3	1.85e-3	2.37e-3
1	8.87e-3	2.88e-3	6.19e-4	1.85e-4	3.37e-4
5	5.16e-3	3.76e-3	8.81e-5	9.79e-5	1.38e-4
10	1.28e-3	6.71e-4	1.88e-5	8.36e-5	8.78e-5
20	2.03e-3	4.28e-4	3.87e-6	5.46e-5	6.87e-5

Table 9 Relative error for the 15 dimensional integral

N	A	t	F	t	S	t	L	t	O	t
10^3	3.16e-3	9.24	5.34e-2	0.08	2.04e-3	0.98	1.06e-2	0.12	7.54e-3	27.4
10^4	1.49e-3	88	1.22e-3	0.93	2.89e-4	9.3	7.33e-3	1.07	6.51e-4	81.5
10^5	5.76e-4	847	3.08e-4	9.65	1.13e-4	93.8	1.54e-3	10.11	7.29e-5	242.1
10^6	1.29e-4	8235	1.37e-5	96.9	1.93e-5	935	1.14e-4	99.6	8.29e-6	720.2

Table 10 Relative error for the 15 dimensional integral

time(s)	A	F	S	L	O
1	9.96e-1	6.30e-3	1.10e-3	3.64e-3	3.51e-2
5	7.72e-2	1.68e-3	2.45e-4	7.32e-4	1.23e-2
10	1.33e-2	5.89e-3	9.48e-5	1.94e-4	9.63e-3
20	3.03e-2	1.66e-3	9.87e-6	4.05e-5	7.51e-3
100	8.11e-3	5.04e-4	8.17e-7	4.03e-6	9.51e-5

Table 11 Relative error for the 25 dimensional integral

N	O	t,s	S	t,s	L	t,s
10^3	3.77e-3	2.03	1.47e-1	0.4	7.54e-1	0.02
10^4	3.18e-3	19.52	5.68e-2	5.64	5.39e-2	0.15
10^5	5.33e-5	181	7.21e-3	33.4	2.11e-2	1.07
$10^6 3$	3.11e-5	1234	2.89e-3	161	1.71e-4	8.21

Table 12 Relative error for the 25 dimensional integral

time,sec	O	S	L
1	7.24e-2	9.51e-2	2.11e-2
5	8.16e-3	5.76e-2	1.61e-2
10	5.18e-3	2.71e-2	9.58e-3
20	3.13e-3	8.28e-3	7.87e-3

Table 13 Relative error for 30 dimensional integral

N	O	t,s	S	t,s	L	t,s
10^3	2.01e-2	5.4	1.18e-1	0.42	8.81e-1	0.02
10^4	6.53e-3	14.5	8.40e-2	4.5	6.19e-2	0.14
10^5	1.35e-3	145	1.18e-2	30.2	2.78e-2	1.16
10^6	2.11e-4	1290	9.20e-3	168	9.86e-3	8.61

Table 14 Relative error for 30 dimensional integral

time,sec	O	S	L
1	4.38e-1	1.01e-1	2.38e-2
5	1.16e-2	7.76e-2	1.81e-2
10	8.11e-3	5.71e-2	9.48e-3
20	4.63e-3	1.28e-2	7.87e-3

dimensions up to 10, the FIBO and Optimal methods achieve the smallest relative error for a given number of samples—see Tables 1, 3, 5. The Optimal method is with 1 order better than the FIBO method for a fixxed number of samples, but the advantage of FIBO method is the significantly lower computational time. It is seen that for dimensions 10 and 15, the Optimal method gives relative errors for a fixed number of samples with at least 1 order better than FIBO and Sobol, see Tables 7 and 9, with FIBO achieving better accuracy in much less time—see Tables 8 and 10. The adaptive algorithm is suitable for higher dimensions because of the small number of samples to achieve the preliminary given relative error. The Adaptive algorithm performs better than FIBO and Sobol—see Tables 11 and 13. Analyzes show that for high 25 and 30 dimensions, the Optimal MC approach outperforms LHS and Sobol's quasi-Monte Carlo algorithm - see Tables 12 and 14, where the LHS method produce better results than the Sobol QMC approach. Here we do not test Adaptive approach, because of the significant amount of time and FIBO method which gives unreliable relative errors for the high dimensional case.

4 Conclusions

An optimal stochastic approach for multidimensional integrals up to 30 dimensions has been presented. The numerical results including relative errors and computational times corresponding to the algorithms under consideration are presented, and the algorithms efficiency is discussed. The efficiency of the optimal Monte Carlo algorithm for the calculation of multidimensional integrals has been presented. The optimal approach appears to be an efficient stochastic solution to solve this kind of problem, because we demonstrate here its reliability for calculating integrals with smooth integrands in a relatively small subregion of the original integration domain regardless of the dimension. The presented optimal MC algorithm is the best optimized stochastic approach for multidimensional integrals of smooth integrand functions with a relatively low dimension.

Acknowledgements Venelin Todorov is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria and by the Bulgarian National Fund of Science under Project DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Computational Problems". Stoyan Apostolov is supported by the Bulgarian National Science Fund under Young Scientists Project KP-06-M32/2 - 17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics". The work is also supported by Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications" funded by National Science Fund - Bulgaria.

References

1. Antonov, I., Saleev, V.: An Economic Method of Computing LP_τ-sequences, USSR Comput. Math. Phy. **19**, 252–256 (1979)
2. Atanassov, E., Dimov, I.T.: A new optimal monte carlo method for calculating integrals of smooth functions. Journal of Monte Carlo Methods and Applications **5**(2), 149–167 (1999)
3. Berntsen, J., Espelid, T.O., Genz, A.: An adaptive algorithm for the approximate calculation of multiple integrals, ACM Trans. Math. Softw. **17**, 437–451 (1991)
4. Bratley, P., Fox, B.: Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator. ACM Transactions on Mathematical Software **14**(1), 88–100 (1988)
5. Cull, P., Holloway, J.L.: Computing Fibonacci numbers quickly. Inf. Process. Lett. **32**(3), 143–149 (1989)
6. Davis P.J. and Rabinowitz P. (1984) Methods of Numerical Integration. Academic Press, London, 2nd edition
7. Dimov I. (2008) Monte Carlo Methods for Applied Scientists, New Jersey, London, Singapore, World Scientific, 291 p., ISBN-10 981-02-2329-3
8. Dimov, I.T., Tonev, O.: Monte carlo algorithms: performance analysis for some computer architectures. J. of Computational and Applied Mathematics **48**, 253–277 (1993)
9. Dupach, V.: Stochasticke pocetni metody. Cas. pro pest. mat. **81**(1), 55–68 (1956)
10. Ermakov, S.M.: Monte Carlo Methods and Mixed Problems. Nauka, Moscow (1985)
11. Filipova, M., I. R. Georgiev, and I. Zheleva. "Numerical study of the influence of the heater position upon the heat transfer during pyrolysis process used for end-of-life tires treatment." AIP Conference Proceedings. Vol. 2025. No. 1. AIP Publishing LLC, 2018
12. Georgiev, I., et al. "Comparison of heuristic algorithms for solving a specific model of transportation problem." AIP Conference Proceedings. Vol. 2302. No. 1. AIP Publishing LLC, 2020
13. Georgieva, R.: PhD Thesis: Computational complexity of Monte Carlo algorithms for multidimensional integrals and integral equations, Sofia, 2003
14. Metropolis, N., Ulam, S.: The Monte Carlo Method. J. of Amer. Statistical Assoc. **44**(247), 335–341 (1949)
15. Minasny, B., McBratneyB, B.: A conditioned Latin hypercube method for sampling in the presence of ancillary information. Journal Computers and Geosciences archive **32**(9), 1378–1388 (2006)
16. Nikolskii, S.: Quadrature Formulas. Nauka, Moscow (1988)
17. Sellier, J.M., Dimov, I.: On a full Monte Carlo approach to quantum mechanics. Physica A **463**, 45–62 (2016)
18. Sloan, I.H., Kachoyan, P.J.: Lattice methods for multiple integration: Theory, error analysis and examples, SIAM. J. Numer. Anal. **24**, 116–128 (1987)
19. Sloan, I.H., Joe, S.: Lattice Methods for Multiple Integration. Oxford University Press, Oxford (1994)
20. Sobol, I.: Numerical methods Monte Carlo. Nauka, Moscow (1973)
21. Sobol, I.M.: "Distribution of points in a cube and approximate evaluation of integrals". Zh. Vych. Mat. Mat. Fiz. 7: 784–802 (in Russian); U.S.S.R Comput. Maths. Math. Phys. **7**, 86–112 (1967)
22. I.M. Sobol, (1989) Quasi-Monte Carlo methods, in: Sendov, Bl., Dimov, I.T. (eds.) International Youth Workshop on Monte Carlo Methods and Parallel Algorithms, World Scientific, Singapore: 75–81
23. Wang Y., F.J. Hickernell (2002) An historical overview of lattice point sets

Expansions on Quadrature Formulas and Numerical Solutions of Ordinary Differential Equations

Venelin Todorov, Yuri Dimitrov, Radan Miryanov, Ivan Dimov, and Stoyan Poryazov

Abstract This paper is a continuation of the discussion on optimization of the quadrature formulas and their applications in paper [6]. Second-order numerical solutions of Volterra integral equations are constructed using the quadrature formulas obtained in [6]. The numerical results presented in the paper confirm the effectiveness of the methods for numerical solution of ordinary differential equations.

1 Introduction

Numerical quadrature is a term used for numerical integration in one dimension. Numerical integration in more than one dimension is usually called cubature. There is a broad class of computational algorithms for numerical integration. The Newton-Cotes quadrature formulas (1) are a group of formulas for numerical integration where the integrand is evaluated at equally spaced points. Let $h = (b - a)/n$.

V. Todorov (✉) · Y. Dimitrov · S. Poryazov
Institute of Mathematics and Informatics, Department of Information Modeling, Bulgarian Academy of Sciences, Acad. Georgi Bonchev Str, Block 8, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

Y. Dimitrov
e-mail: yuri.dimitrov@ltu.bg

S. Poryazov
e-mail: stoyan@math.bas.bg

V. Todorov · I. Dimov
Institute of Information and Communication Technologies, Department of Parallel Algorithms, Bulgarian Academy of Sciences, Acad. G. Bonchev Str, Block 25 A, 1113 Sofia, Bulgaria
e-mail: ivdimov@bas.bg

R. Miryanov
University of Varna, Sofia, Bulgaria
e-mail: stefka@parallel.bas.bg

Y. Dimitrov
Department of Mathematics and Physics, University of Forestry, Sofia, Bulgaria

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_25

$$h\sum_{k=0}^{n} w_k f(b-kh) \approx \int_a^b f(x)dx. \tag{1}$$

The two most popular approximations for the definite integral are the trapezoidal approximation (2) and the midpoint approximation (3).

$$\mathcal{A}_T^h f(x) = \frac{h}{2}\left(f(a) + 2\sum_{k=1}^{n-1} f(a+kh) + f(b)\right) = \int_a^b f(x)dx + O\left(h^2\right), \tag{2}$$

$$\mathcal{A}_M^h f(x) = h\sum_{k=0}^{n-1} f\left(a + (k+1/2)\,h\right) = \int_a^b f(x)dx + O\left(h^2\right). \tag{3}$$

The midpoint approximation and the trapezoidal approximation for the definite integral are Newton-Cotes quadrature formulas with accuracy $O\left(h^2\right)$. The trapezoidal and the midpoint approximations are constructed by dividing the interval $[a, b]$ to subintervals of length h and interpolating the integrand function by Lagrange polynomials of degree zero and one. Another important Newton-Cotes quadrature formula is the Simpson's approximation [4]. The Simpson's approximation has a fourth-order accuracy and is obtained by interpolating the function by a second degree Lagrange polynomial. Denote by $\mathcal{E}_T^h f(x)$ and $\mathcal{E}_M^h f(x)$ the errors of the trapezoidal and the midpoint approximations.

$$\mathcal{E}_T^h f(x) = \mathcal{A}_T^h f(x) - \int_a^b f(x)dx, \quad \mathcal{E}_M^h f(x) = \mathcal{A}_M^h f(x) - \int_a^b f(x)dx.$$

From the Euler-McLaurin formula the error of the trapezoidal approximation on the interval $[a, b]$ is expressed as

$$\mathcal{E}_T^h f(x) = \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right)h^{2k} + R_{2m}^T(h)h^{2m}, \tag{4}$$

$$R_{2m}^T(h) = \frac{1}{(2m)!}\int_a^b \tilde{B}_{2m}(a - t/h)f^{(2m)}(t)dt,$$

where $\tilde{B}_n(x) = B_n(\{x\}) = B_n(x - \lfloor x \rfloor)$ is the Bernoulli 1-periodic function and $B_n(x) = \sum_{k=0}^{n}\binom{n}{k}B_{n-k}x^k$ is the Bernoulli polynomial of degree n. The error of the midpoint approximation satisfies [22]

$$\mathcal{E}_M^h f(x) = \sum_{k=1}^{m} \frac{\left(2^{1-2k} - 1\right)B_{2k}}{(2k)!}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right)h^{2k} + R_{2m}^M(h)h^{2m}, \tag{5}$$

$$R_{2m}^{M}(h) = \frac{1}{(2m)!}\int_a^b \tilde{B}_{2m}((a+b)/2 - t/h) f^{(2m)}(t)dt.$$

The terms $R_{2m}^{T}(h)$ and $R_{2m}^{M}(h)$ satisfy $\lim_{h\to 0} R_{2m}^{T}(h) = \lim_{h\to 0} R_{2m}^{M}(h) = 0$, when the function $f(x) \in C^{2m}[a,b]$. The midpoint approximation is around two times more accurate than the trapezoidal approximation. The Simpson's approximation is constructed as a linear combination of the midpoint approximation and the trapezoidal approximation with weights $1/3$ and $2/3$.

The Gaussian quadrature formulas $\sum_{i=1}^{n} w_i f(x_i)$ are approximations for the definite integral where the integrand is evaluated at unequally spaced points [16]. The Gaussian quadrature formulas are constructed to yield an exact result for the polynomials of degree $2n-1$ by using suitable values of the nodes x_i and weights w_i. The accuracy of the Newton-Cotes and the Gaussian quadrature formulas is lower when the function $f(x)$ has singularities. The Monte Carlo methods for numerical integration are computational algorithms where the nodes are chosen as random, pseudo-random or quasi-random numbers in the area of integration [9]. The Monte Carlo methods are suitable for approximation of multidimensional integrals [14, 15].

The generating function of quadrature formula (1) is defined as

$$G(x) = \sum_{k=0}^{\infty} w_k x^k.$$

The Riemann sum approximation for the definite integral is related to the trapezoidal approximation and has a generating function $1/(1-x)$. The Riemann sum approximation for the fractional integral of order α has a generating function $Li_{1-\alpha}(x)$, where $Li_{1-\alpha}(x)$ is the polylogarithm function of order $1-\alpha$. The midpoint approximation is a shifted approximation with shift parameter $h/2$. The generating function of the midpoint approximation is $\sqrt{x}/(1-x)$. Methods for approximation of fractional integrals and derivatives based on the Fourier transform and the generating function of the approximation have been studied by Chen and Deng [3], Ding and Li [11, 12], Lubich [23], Tuan and Gorenflo [25]. In [8] we use the series expansion formula of the polylogarithm function $Li_{1-\alpha}(e^x)$, to derive the expansion formula of the trapezoidal approximation for the fractional integral of order α. In the present paper we construct second, third and fourth order approximations for the definite integral with generating functions $G_1(x) = \pi \sec(\pi\sqrt{x}/2)/4$ and $G_2(x) = \pi \tan(\pi\sqrt{x}/2)/(4\sqrt{x})$. The two generating functions $G_1(x)$ and $G_2(x)$ are positive and increasing functions and have a vertical asymptote at the point $x = 1$.

The graphs of the functions $G_1(x)$ and $G_2(x)$ are given in Fig. 1.

The tangent and secant functions have McLaurin series

$$\sec x = \sum_{k=0}^{\infty} \frac{(-1)^k E_{2k}}{(2k)!} x^{2k}, \tag{6}$$

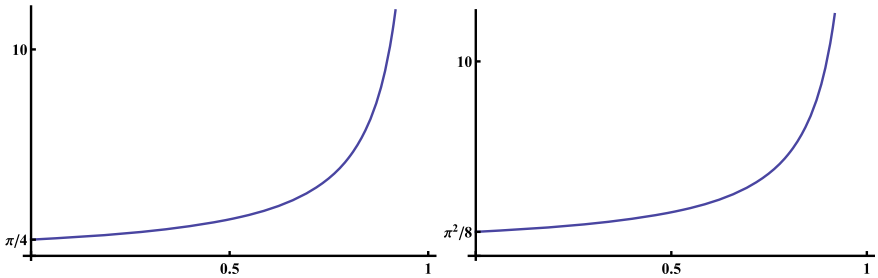


Fig. 1 Graphs of the generating functions $G_1(x) = \pi \sec(\pi\sqrt{x}/2)/4$ (left) and $G_2(x) = \pi \tan(\pi\sqrt{x}/2)/(4\sqrt{x})$ (right)

$$\tan x = \sum_{k=0}^{\infty} \frac{(-4)^{k+1}\left(1-4^{k+1}\right) B_{2k+2}}{(2k+2)!} x^{2k+1}. \tag{7}$$

The radius of convergence of power series (6) and (7) is $|x| < \pi/2$. The first eight elements of the sequences of Euler and Bernoulli numbers are $1, 0, -1, 0, 5, 0, -61, 0, \cdots$ and $1, -1/2,\ 1/6, 0, -1/30, 0, 1/42, 0, \cdots$. The Euler and Bernoulli numbers with an odd index are equal to zero. The sequence of Euler numbers $\{E_{2n}\}$ is an alternating sequence of integers, and $\{B_{2n}\}$ is an alternating sequence of rational numbers.

Efficient methods for computation of the Euler numbers, Bernoulli numbers and the series expansions of the secant and tangent functions are discussed by Atkinson [1], Brent and Harvey [2], Knuth and Buckholtz [21]. The values of the Euler numbers, Bernoulli numbers and Riemann zeta function are included in the modern software packages for scientific computing. Denote by $\bar{E}_k$ and $\bar{B}_k$ the coefficients of the McLaurin series expansion of the generating functions $G_1(x)$ and $G_2(x)$.

$$\bar{E}_k = \frac{|E_{2k}|}{(2k)!}\left(\frac{\pi}{2}\right)^{2k+1}, \quad \bar{B}_k = \frac{\left(4^{k+1}-1\right)\pi^{2k+2}\,|B_{2k+2}|}{(2k+2)!}.$$

2 Problem Settings

In paper [6] we study the quadrature formulas which have generating functions $G_1(x) = \pi \sec(\pi\sqrt{x}/2)/4$ and $G_2(x) = \pi \tan(\pi\sqrt{x}/2)/(4\sqrt{x})$. We construct the second-order quadrature formulas

$$\frac{h}{2}\left(y_0 + \sum_{k=1}^{N-1} \overline{E}_k y_{N-k} + \frac{\pi-1}{2} y_N\right) = \int_a^b y(x)dx + O\left(h^2\right),$$

$$\frac{h}{2}\left(y_0+\sum_{k=1}^{N-1}\overline{B}_k y_{N-k}+\frac{\pi^2-6}{4}y_N\right)=\int_a^b y(x)dx+O\left(h^2\right),$$

where $h=(b-a)/N$ and $x_k=a+kh,\ y_k=y(x_k)$ for $k=0,1,\cdots,N$ and $\overline{E}_k$ and $\overline{B}_k$ are the coefficients of the Maclaurin series of the generating functions

$$\overline{E}_k=\frac{|E_{2k}|}{(2k)!}\left(\frac{\pi}{2}\right)^{2k+1},\ \overline{B}_k=\frac{(4^{k+1}-1)\pi^{2k+2}|B_{2k+2}|}{(2k+2)!}, \tag{8}$$

where E_k and B_k are the Euler and Bernoulli numbers. The coefficients of the right endpoint expansion formulas are equal to the coefficients of the Maclaurin series of the functions $H_i(x)=G_i(e^{-x})$ for $i=1,2$. In [6] we construct third-order and fourth-order quadrature formulas as linear combinations of the trapezoidal rule

$$\frac{h}{2}\left(y_0+2\sum_{k=1}^{N-1}y_{N-k}+y_N\right)=\int_a^b y(x)dx+O\left(h^2\right), \tag{9}$$

the two quadrature formulas above and their modifications. The method for construction of quadrature formulas by first specifying the generating function is an effective method for construction of approximations of the fractional derivatives and integrals (see [5, 7, 10, 13]). The method is applicable for construction of approximations of the definite integral and the integer order derivatives of a function as well. In [18] we construct approximations of the first derivative which have exponential and logarithmic generating functions. In the paper we give a proof for the convergence of the approximations and applications of the approximations for numerical solution of ordinary and partial differential equations. The method used in the paper can be extended for construction of approximations of the second derivative which are suitable for deriving approximations of the fractional derivatives. Other methods for numerical solution of integral equations use Gaussian quadratures on non-uniform nets and Monte Carlo methods for numerical integration (see [19, 20, 26]). Let

$$\frac{h}{2}\sum_{i=0}^{N}w_i y_{N-i}=\int_a^b y(t)dt+O\left(h^2\right) \tag{$*$}$$

be a second-order quadrature. Consider a Voltera integral equation of the second kind which has the following form

$$y(x)+\int_0^x K(x-t)y(t)dt=F(x). \tag{10}$$

The Nyström method (see [17]) for computing the numerical solution of equation (10) uses the approximations of the values of the definite integral in equation (10) with quadrature formula (2) on all intervals $[0,x_n]$,

Table 1 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	4.545×10^{-6}	2.000	2.802×10^{-6}	2.000
0.0025	1.136×10^{-6}	2.000	7.006×10^{-7}	2.000
0.00125	2.841×10^{-7}	2.000	1.751×10^{-7}	2.000

$$y_n + \frac{h}{2}\sum_{i=0}^{n} w_i K_i y_{n-i} = F_n + O\left(h^2\right).$$

The numerical solution $\{u_n\}_{n=0}^{N}$ of integral equation (10), where u_n is an approximation of the value of the solution y_n, is computed recursively with

$$u_n = \frac{1}{2 + w_0 K_0 h}\left(2F_n - h\sum_{i=1}^{n} w_i K_i y_{n-i}\right) \qquad (NS1(*))$$

and has an initial condition $u_0 = y_0 = F(0)$. The computation of numerical solution NS(*) involves $O\left(N^2\right)$ operations. Denote by (3.1) and (3.2) the equations obtained from equation (3) with a kernel function $K(s) = 3 + 2s$ and right hand side $F_1(x) = x^3(10(4x^2 + 30x + 40)\ln x - 18x^2 - 75x)/400$ and

$$F_2(x) = \arctan x - x + (3/2 + x)\ln(1 + x^2) + (1 + 3x + x^2)\operatorname{arccot} x$$

respectively. Equations (3.1) and (3.2) have the solutions $y(x) = x^3 \ln x$ and $y(x) = \operatorname{arccot} x$. In Table 1 we give the experimental results for the error and the order of the numerical solution NS(9), which uses the trapezoidal rule (9), of equations (3.1)-left and (3.2)-right on the interval [0, 1]. The rest of the paper is organized as follows. In section two we construct the numerical solution of first order ODEs by transforming them to integral equations in the form (3) and we also construct the numerical solutions which use the corresponding shifted quadrature formulas. In section three of the paper we apply the method for numerical solution of second order ordinary differential equations (ODEs) by converting them to Voltera integral equations.

3 Numerical Solution of First Order ODEs

In this section we construct the numerical solution of first order ODEs by first transforming them to integral equations which are solved with Niström method. In [6] we obtain the second order approximations

$$\frac{h}{2}\sum_{k=0}^{N}\overline{E}_k y_{N-k} = \int_a^b y(x)dx + O\left(h^2\right), \tag{11}$$

$$\frac{h}{2}\sum_{k=0}^{N}\overline{B}_k y_{N-k} = \int_a^b y(x)dx + O\left(h^2\right), \tag{12}$$

where $\overline{E}_0 = (\pi - 1)/2$, $\overline{B}_0 = (\pi^2 - 6)/4$ and the rest of the weights are defined with (8). The two approximations (11) and (12) require that the integrand function satisfies $y(a) = 0$. We use the method (see [7, 18]) for extending the approximations to the class of all differentiable functions by changing the values of the last weights. By applying approximation (11) to the function $y(x) - y(a)$ we obtain

$$\frac{h}{2}\sum_{k=0}^{N-1}\overline{E}_k(y_{N-k} - y_0) = \int_a^b (y(x) - y(a))dx + O\left(h^2\right),$$

$$\frac{h}{2}\left(\sum_{k=0}^{N-1}\overline{E}_k y_{N-k} + 2Ny_0 - y_0\sum_{k=0}^{N-1}\overline{E}_k\right) = \int_a^b y(x)dx + O\left(h^2\right).$$

Define $\overline{E}_N = 2N - \sum_{k=0}^{N-1}\overline{E}_k$. Therefore approximation (11) has a second-order accuracy for all differentiable functions. Similarly, when $\overline{B}_N = 2N - \sum_{k=0}^{N-1}\overline{B}_k$ approximation (12) holds for all differentiable functions in $[a, b]$. Now we apply approximations (11) and (12) for numerical solution of ODEs. Consider the first order linear ODE

$$y' + ay = f(x), \;\; y(0) = y_0. \tag{13}$$

By applying integration on both sides of equation (13) we get

$$\int_0^x y'(t)dt + a\int_0^x y(t)dt = \int_0^x f(t)dt,$$

$$y(x) + a\int_0^x y(t)dt = F(x), \tag{14}$$

where $F(x) = y_0 + \int_0^x y(t)dt$. Equation (14) is equivalent to (13) and it is a Voltera integral equation of the second kind with a kernel $K(s) = a$. Denote by (7.1) the equation which corresponds to (14) when $a = 2$ and $F_1(x) = x^3(4(2 + x)\ln x - x)/8$. Equation (7.1) has the solution $y(x) = x^3 \ln x$ and it is equivalent to equation (13) with a right hand side $f_1(x) = x^2(1 + (3 + 2x)\ln x)$. Denote by (7.2) the integral equation which corresponds to equation (14) with $a = 3$, $F_2(x) = (1 + 3x)\operatorname{arccot} x + 3\ln(1 + x^2)/2$ and initial condition $y(0) = y_0 = F_0(0) = \pi/2$. Equation (7.2) has the solution $y(x) = \arccos x$ and it is equivalent to equation (13) with a right hand side $f_2(x) = 3\operatorname{arccot} x - 1/(1 + x^2)$. Equations (7.1) and (7.2)

Table 2 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	7.202×10^{-6}	1.995	2.918×10^{-6}	1.969
0.0025	1.804×10^{-6}	1.997	7.412×10^{-7}	1.977
0.00125	4.513×10^{-7}	1.999	1.870×10^{-7}	1.987

Table 3 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	3.823×10^{-6}	2.004	1.351×10^{-6}	1.996
0.0025	9.544×10^{-7}	2.002	3.385×10^{-7}	1.997
0.00125	2.384×10^{-7}	2.001	8.476×10^{-8}	1.998

are integral equations in the form (3) and can be solved numerically with methods NS(11) and NS(12), which use quadrature formulas (11) and (12). In Table 2 we give the experimental results for the error and the order of the numerical solution NS(11) of equations (7.2) and (7.3).

The results for the error and order of the numerical solution NS(12) of equations (7.2) and (7.3) are given in Table 3.

Now we construct the shifted quadratures which correspond to formulas (11) and (12). From [6]

$$\frac{h}{2}\sum_{k=0}^{N}\overline{E}_k y_{N-k} = \int_a^b y(x)dx + \frac{1}{4}f(b)h + O\left(h^2\right), \tag{15}$$

$$\frac{h}{2}\sum_{k=0}^{N}\overline{B}_k y_{N-k} = \int_a^b y(x)dx + \frac{3}{4}f(b)h + O\left(h^2\right), \tag{16}$$

where the weights $\overline{E}_k$ and $\overline{B}_k$ are defined with (8) for all indices $k = 0, 1, \cdots, n$. From the mean value theorem we have the second order approximation

$$\int_a^{b+ch} y(x)dx = \int_a^b y(x)dx + cf(b)h + O\left(h^2\right). \tag{17}$$

From (15) and (17)

$$\frac{h}{2}\sum_{k=0}^{N}\overline{E}_k y_{N-k} = \int_a^{b+h/4} y(x)dx + O\left(h^2\right). \tag{18}$$

Shifted quadature (18) has a requirement that the integrand function satisfies the condition $y(a) = 0$. By applying (18) to the function $y(x) - y(a)$ we find

$$\frac{h}{2}\sum_{k=0}^{N-1}\overline{E}_k\left(y_{N-k} - y_0\right) = \int_a^{b+h/4}(y(x) - y(a))dx + O\left(h^2\right),$$

Shifted quadrature (18) has a second order accuracy for all differentiable functions when the weight $\overline{E}_N$ is defined as

$$\overline{E}_N = \frac{2}{h}\left(b - a + \frac{h}{4}\right) - \sum_{k=0}^{N-1}\overline{E}_k = 2N + \frac{1}{2} - \sum_{k=0}^{N-1}\overline{E}_k$$

Similarly from (16) and (17) we obtain

$$\frac{h}{2}\sum_{k=0}^{N}\overline{B}_k y_{N-k} = \int_a^{b+3h/4} y(x)dx + O\left(h^2\right). \tag{19}$$

Shifted quadrature (19) has a second-order accuracy when

$$\overline{B}_N = 2N + \frac{3}{2} - \sum_{k=0}^{N-1}\overline{B}_k$$

and the weights $\overline{B}_0, \overline{B}_1, \cdots, \overline{B}_{N-1}$ are defined with (8). The first weights $\overline{E}_0 = \pi/2$ and $\overline{B}_0 = \pi^2/4$. Now we construct the numerical solution of integral equation (14) which uses shifted quadrature (18). By approximating the definite integral in (14) at the point $x_{n+1/4}$ with (18) we obtain

$$y_{n+1/4} + \frac{1}{2}ah\sum_{k=0}^{n-1}\overline{E}_k y_{n-k} = F_{n+1/4} + O\left(h^2\right).$$

Let $\{u_n\}_{n=0}^N$ be the numerical solution of (14). From the second order approximation

$$y_{n+1/4} = \frac{5y_n - y_{n-1}}{4} + O\left(h^2\right)$$

we obtain the recursive formula for the numerical solution

$$u_n = \frac{1}{5 + \pi ah}\left(4F_{n+1/4} + u_{n-1} - 2ah\sum_{k=1}^{n-1}\overline{E}_k y_{n-k}\right), \tag{NS1}$$

Table 4 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	5.638×10^{-6}	2.000	1.323×10^{-6}	1.996
0.0025	1.409×10^{-6}	2.000	3.314×10^{-7}	1.998
0.00125	3.522×10^{-7}	2.000	8.291×10^{-8}	1.989

Table 5 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	6.249×10^{-5}	1.990	1.352×10^{-5}	1.900
0.0025	1.567×10^{-5}	1.995	3.532×10^{-6}	1.936
0.00125	3.924×10^{-6}	1.997	9.054×10^{-7}	1.964

with an initial condition $u_0 = F_0$. In Table 4 we give the experimental results for the error and the order of the numerical solution NS1 of Eqs. (7.2)-left and (7.3)-right. The numerical solution of integral equation (14) which uses shifted quadrature (19) is obtained from the approximation of the definite integral in (14) at the point $x_{n+3/4}$ with (19).

$$y_{n+3/4} + \frac{1}{2}ah\sum_{k=0}^{n-1}\overline{B}_k y_{n-k} = F_{n+3/4} + O\left(h^2\right).$$

From

$$y_{n+3/4} = \frac{7y_n - 3y_{n-1}}{4} + O\left(h^2\right)$$

we obtain the recursive formula of the numerical solution

$$u_n = \frac{2}{14 + a\pi^2 h}\left(4F_{n+3/4} + 3u_{n-1} - 2ah\sum_{k=1}^{n-1}\overline{E}_k y_{n-k}\right). \qquad (NS2)$$

The experimental results for the error and the order of the numerical solution NS2 of equations (7.2) and (7.3) are given in table 5.

4 Numerical Solution of Second Order ODEs

We apply the method from Sect. 2 for computing the numerical solution of the second order ODE

$$y'' + 3y' + 2y = f(x), \;\; y(0) = y_0, \, y'(0) = y_0'. \qquad (20)$$

Table 6 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	6.730×10^{-6}	1.997	5.885×10^{-6}	1.947
0.0025	1.684×10^{-6}	1.998	1.499×10^{-7}	1.972
0.00125	4.213×10^{-7}	1.999	3.805×10^{-7}	1.978

Equation (20) is transformed to an integral equation in the form (3) by applying successive integration to both sides

$$\int_0^x y''(u)du + 3\int_0^x y'(u)du + 2\int_0^x y(u)du = \int_0^x f(u)du,$$

$$y'(x) + 3y(x) + 2\int_0^x y(u)du = 3y_0 + y_0' + \int_0^x f(u)du.$$

Integrate again both sides

$$y(x) + 3\int_0^x y(t)dt + 2\int_0^x \int_0^t f(u)dudt = F(x), \tag{21}$$

where $F(x) = y_0 + (3y_0 + y_0')x + \int_0^x \int_0^t f(u)dudt$. By changing the order of integration of the double integral we get

$$\int_0^x \int_0^t f(u)dudt = \int_0^x \int_u^x f(u)dtdu = \int_0^x (x-u)f(u)dtdu.$$

Equation (21) is transformed to

$$y(x) + \int_0^x (3 + 2x - 2t)y(t)dt = F(x). \tag{22}$$

Equation (22) is a Voltera integral equation of the second kind with a kernel $K(s) = 3 + 2s$. In Sect. 1 we compute the numerical solution NS(9) of equations (3.1) and (3.2), which are also integral equations in the form (22). In Table 6 we give the results for the maximal error and order of numerical solution NS(11), which uses quadrature formula (11), of Eqs. (3.1)-left and (3.2)-right and step sizes $h = 0.005, 0.0025, 0.00125$.

The results for the error and the order of the numerical solution NS(12) of equations (3.1) and (3.2) are given in Table 7.

Table 7 The results for the error and the order of the numerical solution

h	$Error$	$Order$	$Error$	$Order$
0.005	3.563×10^{-6}	2.002	2.795×10^{-6}	2.008
0.0025	8.901×10^{-7}	2.001	6.969×10^{-7}	2.004
0.00125	2.224×10^{-7}	2.000	1.740×10^{-7}	2.001

5 Conclusion

In the paper we construct second-order numerical solutions of ordinary differential equations by converting them to integral equations and applying the quadrature formulas from [6] . Numerical solutions NS(11) and NS(12) involve additional multiplications compared to the standard method NS(9) and have a longer computational time. One advantage of the methods discussed in the paper is that the linear ODEs are equivalent to Voltera integral equations in the form (3) and they can be solved numerically with NS($*$) which uses an appropriate quadrature. All methods discussed in the paper involve $O\left(N^2\right)$ computations and have comparable performance. From the experiments presented in the paper and the results of additional experiments we can conclude that the numerical solutions discussed in the paper have a stable and efficient performance. In future work we will prove the convergence of the numerical solutions constructed in the paper and we will apply the methods for numerical solution of other classes of ordinary differential equations.

Acknowledgements Venelin Todorov is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICT in SES)", contract No DO1-205/23.11.2018, financed by the Ministry of Education and Science in Bulgaria and by the National Scientific Program for post doctoral and young scientists of Ministry of Education and Science 2020-2021. Yuri Dimitrov is supported by the Bulgarian National Science Fund under Young Scientists Project KP-06-M32/2 - 17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics". The work is also supported by the Bulgarian National Fund of Science under Project DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Computational Problems" and by Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications" funded by National Science Fund - Bulgaria.

References

1. Atkinson, M.D.: How to compute the series expansions of sec x and tan x. Amer. Math. Monthly **93**, 387–388 (1986). https://doi.org/10.2307/2323604
2. Brent, R.P., Harvey, D.: Fast computation of Bernoulli, tangent and secant numbers. In: Proceedings of a Workshop on Computational and Analytical Mathematics in honour of Jonathan Borwein's 60th birthday, Springer Proceedings in Mathematics and Statistics, vol. 50, pp. 127–142 (2013)

3. Chen, M., Deng, W.: A second-order accurate numerical method for the space-time tempered fractional diffusion-wave equation. arXiv:1610.02661 (2016)
4. Davis, P.J., Rabinowitz, P.: Methods of numerical integration. 2nd edn. Academic (1984)
5. Dimitrov, Y.: Approximations for the Caputo derivative (I). J. Fract. Calc. Appl. **9**(1), 35–63 (2018)
6. Dimitrov, Y., Miryanov, R., Todorov, V.: Quadrature formulas and Taylor series of secant and tangent. Econ. Comput. Sci. **4**, 23–40 (2017)
7. Dimitrov, Y., Miryanov, R., Todorov, V.: Asymptotic expansions and approximations of the Caputo derivative. Comput. Appl. Math. **37**(4), 5476–5499 (2018)
8. Dimitrov, Y.: Higher-order numerical solutions of the fractional relaxation-oscillation equation using fractional integration. arXiv:1603.08733 (2016)
9. Dimov, I.: Monte Carlo Methods for Applied Scientists. World Scientific (2008)
10. Ding, H., Li, C.: High-order numerical algorithms for Riesz derivatives via constructing new generating functions. J. Sci. Comput. **71**(3), 759–784 (2017)
11. Ding, H., Li, C.: High-order numerical algorithms for Riesz derivatives via constructing new generating functions. J. Sci. Comput. (2016)
12. Ding, H., Li, C.: High-order algorithms for Riesz derivative and their applications (III). Fract. Calc. Appl. Anal. **19**(1), 19–55 (2016)
13. Gao, G.H., Sun, Z.Z., Zhang, H.W.: A new fractional numerical differentiation formula to approximate the Caputo fractional derivative and its applications. J. Comput. Phys. **259**, 33–50 (2014)
14. Georgiev, I. et al.: Comparison of heuristic algorithms for solving a specific model of transportation problem. In: AIP Conference Proceedings, vol. 2302. No. 1. AIP Publishing LLC (2020)
15. Georgieva, R.: Computational complexity of Monte Carlo algorithms for multidimensional integrals and integral equations. Ph.D. thesis, BAS (2009)
16. Gil, A., Segura, J., Temme, N.M.: Numerical methods for special functions. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (2007)
17. Nyström, E.J.: Über die praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. Acta Math. **54**(1), 185–204 (1930)
18. Todorov, V., Dimitrov, Y., Dimov, I.: Second order shifted approximations for the first derivative. In: Proceedings of International Conference HPC 2019 Studies in Computational Intelligence, Borovetz, Bulgaria (2019)
19. Farnoosh, R., Ebrahimi, M.: Monte Carlo method for solving Fredholm integral equations of the second kind. Appl. Math. Comput. **195**, 309–315 (2008)
20. Kim, S.: Solving singular integral equations using Gaussian quadrature and overdetermined system. Comput. Math. Appl. **35**(10), 63–71 (1998)
21. Knuth, D.E., Buckholtz, T.J.: Computation of tangent, Euler and Bernoulli numbers. Math. Comput. **21**, 663–688 (1967)
22. Kouba, O.: Bernoulli polynomials and applications. Lecture Notes (2013)
23. Lubich, C.: Discretized fractional calculus. SIAM J. Math. Anal. **17**, 704–719 (1986)
24. Tian, W.Y., Zhou, H., Deng, W.H.: A class of second order difference approximations for solving space fractional diffusion equations. Math. Comput. **84**, 1703–1727 (2015)
25. Tuan, V.K., Gorenflo, R.: Extrapolation to the limit for numerical fractional differentiation Z. Agnew. Math. Mech. **75**, 646–648 (1995). https://doi.org/10.1002/zamm.19950750826
26. Zaizai, Y., Zhimin, H.: Using the Monte Carlo method to solve integral equations using a modified control variate. Appl. Math. Comput. **242**, 764–777 (2014)

Research of the Use of Battery Shunting Locomotive with Regenerative Brake

Tsvetomir Gotsov and Venelin Todorov

Abstract Shunting locomotives perform specific functions characterized by short mileages, variable load weights and a high idling rate. Battery technology based on traction is a very suitable solution for shunting locomotives due to the possibility of accumulating kinetic energy in the power supply. The energy efficiency of regenerative braking and the possibilities for efficient shunting in industrial plant were studied.

1 Introduction

Maneuvering activity is a major operation performed on the territory of freight yards, in passengers terminal stations, depots, in industrial plants connected to rail network, in harbours and multimodal terminals. Industrial sites connected to the rail network own shunter locomotives for freight wagons shunting on their premises. In order to be able to operate independently from the overhead voltage, or on non-electrified rail, most shunter locomotives are diesel electric or diesel hydraulic locomotives [1]. Diesel engines typically operate in an inefficient mode, which in a competitive business environment makes them economically unprofitable. To a great extent, this is aided by the rising fuel prices over the past decade and the high maintenance costs of conventional engines. In the case of battery-powered locomotives, these costs are reduced to the maintenance costs of the used battery. Machines on batteries have the possibility of regenerative braking, which enables the kinetic energy to be transformed into electric and re-accumulated in the battery [12]. Interesting is the

T. Gotsov (✉)
Electronics Department, University of Ruse, Ruse, Bulgaria

V. Todorov
Institute of Information and Communication Technologies, Department of Parallel Algorithms, Bulgarian Academy of Sciences, Acad. G. Bonchev Str, Block 25 A, 1113 Sofia, Bulgaria
e-mail: vtodorov@math.bas.bg; venelin@parallel.bas.bg

Institute of Mathematics and Informatics, Department of Information Modeling, Bulgarian Academy of Sciences, Acad. Georgi Bonchev Str, Block 8, 1113 Sofia, Bulgaria

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*,
Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9_26

energy efficiency of this type of braking and its benefits for shunting locomotives. Global warming and pollution issues in general are calling for increasing interest in increasing the efficiency of transport systems, which still account for around 30% of CO_2 sources in Europe and in all industrialized countries [27]. Other important aspect that pushes for strong energy optimization of shunting activities is the liberalization process in both the energy and transport markets. The use of regenerative braking instead of conventional braking makes it possible to convert the relevant part of the train's kinetic energy into electrical power without being distributed on the friction surfaces of the pneumatic brakes, with another important advantage in terms of brake maintenance costs. In addition, reduced wear and extended life of pneumatic pads and discs can lead to significant pollution benefits, as brake wear produces particulate matter and contaminants whose environmental impact is still a matter of monitoring and research.

According to [29] and the studies carried out, it is shown that higher efficiency in regenerative braking is achieved on trains with a rapid braking mode, such as passenger trains, in front of the variant of high-speed trains with high kinetic energy. The main interest is mano-transport locomotives, as they have the highest start-up and braking frequency compared to other types of railway vehicles.

2 Operation of Shunting Locomotives

There is clearly a very great diversity of shunting duties depending from local railways configuration and practices. But it is possible to identify some common characteristics:

- Shunting is operated at low speed, typically below 15 km/h.
- The distance traveled with the shunted vehicle or train is relatively short less than 5 km
- The loads to shunt vary in a wide range from the mass of a shunting locomotive to several thousand tons.

When maneuvering with train braking or unloading special types of wagons, large amounts of compressed air are required, which requires the use of powerful air compressors in this type of locomotive. A special feature of the shunting movements is that the locomotive does not leave its place of work and in a large percentage of cases it has access to standard industrial voltage. This feature of shunting locomotives allows the use of batteries tailored to a single shift for a particular application and a minimum volume and power of the locomotive. The power and mass requirements for the locomotive strongly depend on the terrain on which the locomotive works and the mass of the trains it shunting. For industrial branches of quarries, mines and bulk cargoes, trains usually are up to $1000 tons$ in bulk. Minimal tractive effort for train towing is in the range of $30 - 50kN$ for leveled platforms and about $100kN$ for a slope of 0.5%. Fig. 1 is a mechanical representation of a ES1000 battery locomotive which is the object of the study. Tractive effort is shown in different friction coefficients μ on

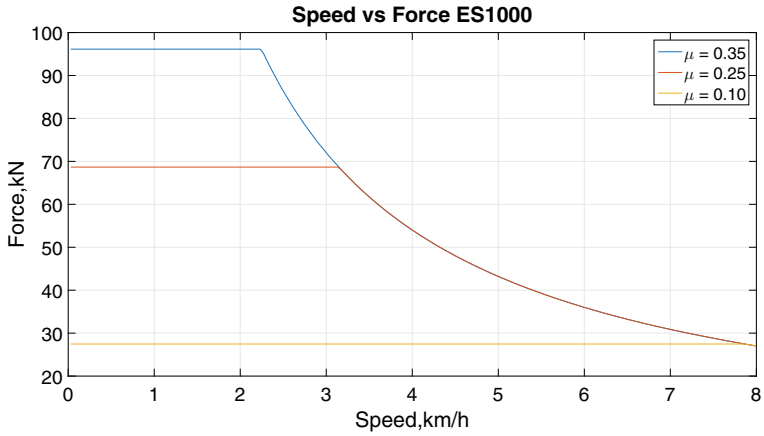


Fig. 1 Traction characteristics ES1000 Shunting locomotive

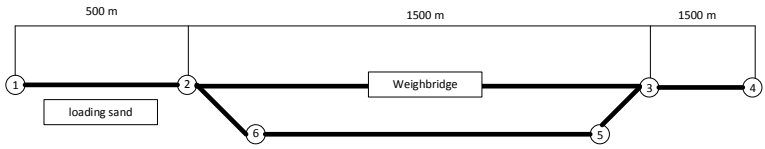


Fig. 2 Railway from industrial plant

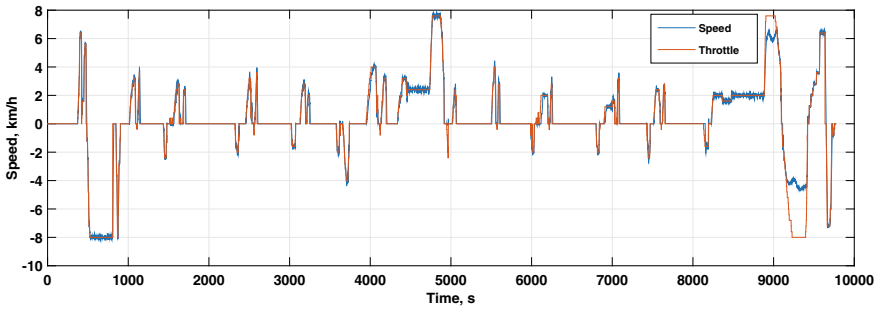


Fig. 3 Typical speed mode when loading 12 freight wagons with chamotte in industrial plant

Fig. 1. For normal dry rails, it is acceptable to use $\mu = 0.35$. For wet rails $\mu = 0.25$ and the worst mode for oiled or frozen rails $\mu = 0.1$.

The track development of an industrial plant for loading chamotte shows Fig. 2. In Sects. 1–2 carriage of wagons is carried out by laying on each wagon.

The mode at which this task is performed can be seen in Fig. 3 in the range 1000–8000 s. After filling all the wagons, the composition passes through the Weighbridge in Sects. 2–3 and interval 8000–9000. Fig. 3. Continue to Sect. 3–4 for a change of the rail. In Sect. 5 there is a stay of full wagons, ready to travel on national infrastructure.

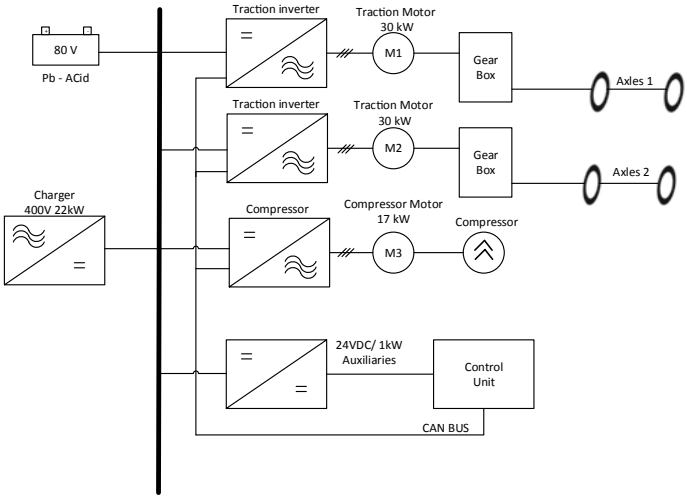


Fig. 4 Electrical diagram from battery shunting locomotive

After the interval 8000 s. In Fig. 3 is the hardest job of the locomotive in which it draws $960t$. The diagram in Fig. 3 shows: in orange the set speed reference and in blue the current engine speed for the entire period of the study. Negative speed marks a backward movement.

3 The System Under Study

Shunting locomotives differ from highway locomotives with the lower power and maximum speed they possess. Battery Shunting locomotives in turn differ with extremely low power due to technological disadvantages of modern sources for energy storage, namely, low density and massive mass. Schematic diagram of an ES1000 shunting locomotive represented in Fig. 4. The locomotive has two axles and has a total weight of 28 t. Each wheel is powered by Induction Motor with a power of 30 kW. There is also an Induction Motors of 17 kW to drive the air compressor. Control of the electric motors is carried out by a four quadrant vector control. As a power source, Lead-Acid Battery with a capacity of $1356Ah$ and a voltage of 80 VDC is used. The total stored power of the locomotive equals $390MJ$, sufficient to provide energy for one work shift. Lead-Acid Battery is the oldest electric energy stored technology. A battery consists of multiple electrochemical cells, connected in parallel and series to form a unit. Cells consist of two electrodes immersed in an electrolyte solution. Batteries work based on the following principle: due to reversible chemical reactions that occur at the electrodes, a potential difference appears between them. Consequently, energy can reversibly change from electrical form to the chem-

Fig. 5 ES1000 in working conditions [2]

ical form [4, 14, 15]. The Lead-Acid battery is unpretentious to the charge current and for short periods can resist larger circuit currents. Charge of the battery from the power grid takes 5 hours, from a fully discharged battery.

Three main types are used in modern battery locomotives: P b-acid, nickel - metal hydride, LiFeO4. The advantage of Lead-Acid battery is the high-rate discharge capability, good specific energy, lower temperature performance, lower thermal management requirement, and low-cost in both manufacturingand recycling are the advantages of the rechargeable battery.

The control is carried out by the Control Unit and the distributed system from the traction controllers. All electronics device connected in CAN-bus network. Data is recorded by an additional installed logger on the shunting locomotive with a sampling time of 0.5 s. Data recording begins immediately after the locomotive exits the Depot for the interval 0–1000 s/Fig. 3/and ends at Depot's place for charge. Locomotive in the experimental conditions of an industrial plant shows in Fig. 5.

4 Results

The experiment was carried out in an industrial plant for loading chamotte. Figure 6 shows a current log consumed by $M1$ for one working cycle ($t \approx 10000\,s.$). The orange graph is the consumed current, and the blue one is the current returned to the battery, marked with a reverse sign. In the conducted trials, the train stop is performed

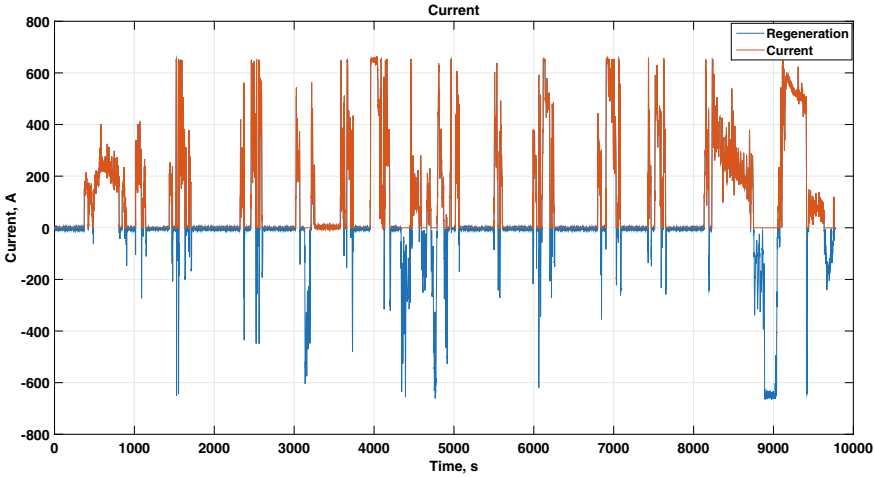


Fig. 6 Traction current log from traction inverter 1

only by a traction engine. The train brake is switched on but not in use. Its goal is to increase the security of shunting.

The stored kinetic energy E_k during movement with a 960 t train and a speed of 8 km/h is:

$$E_k = \frac{mv^2}{2} = \frac{960\,\mathrm{t}(8\,\mathrm{km/h})^2}{2} = 4,7\,\mathrm{MJ}$$

In case of braking, this energy must be scattered into the atmosphere in the form of heat or re-accumulated in the form of electrical energy. Modern propulsion technologies enable much of this energy to be absorbed and reused. These capabilities of electric locomotives and especially battery locomotives create the opportunity to increase energy efficiency in rail transport. The change in battery voltage is illustrated in Fig. 7. The nominal voltage is $80V$ but due to the large discharge currents, it drops to 55 V at discharge and increases to 95 V in the regeneration brake and charge from the Traction inverter.

The study helps us to observe the energy consumed in numerical values by the locomotive and the energy returned to the battery as follows.

The energy consumed without regenerative braking for the shunting cycle:

$$E_{brake} = \int_0^T u(t)i(t)dt = 85.5\,\mathrm{MJ}$$

The energy consumed with regenerative braking for the shunting cycle:

$$E_{regbrake} = \int_0^T u(t)i(t)dt = 60.7\,\mathrm{MJ}$$

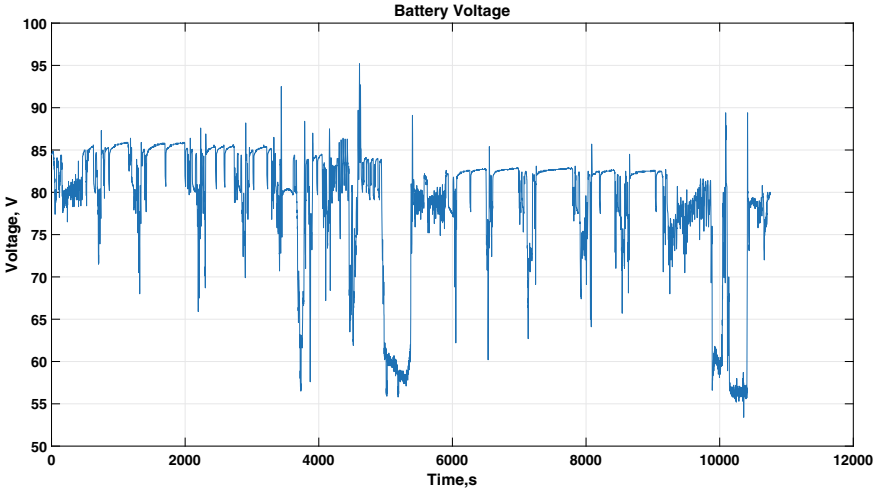


Fig. 7 Voltage drop for Lead-Acid battery

Energy efficiency from the use of regenerative braking:

$$EE = \left(\frac{E_{brake}}{E_{regbrake}} - 1 \right) * 100 = 41\%$$

The research is done from recorded data on one of the traction inverters. The total power consumption of the two drives and the air compressor is $219MJ$ $(61kWh)$ measured as the electricity required to charge the battery from the power grid. For the same amount of work done with a diesel locomotive Class 52, the energy consumed equals 80 liters of diesel fuel. The data for about 41% efficiency is also confirmed by other publications related to the study of regenerative braking energy efficiency [3–8, 17–26]. The one-year electricity savings during similar work operations from the use of a regenerative brake is approximately:

$$E_{reg} = E_{brake} - E_{regbrake} = 6000\,\mathrm{MJ} = 1.6\,\mathrm{MW/h}$$

The economic benefit of using a battery shunting locomotive instead of manoeuvrable diesel engine is the electricity consumption for a year /instead of diesel fuel/:

$$E_{total} = (240 workingdaysright)(61\,\mathrm{kW/h}) = 14640\,\mathrm{kW/h}$$

Diesel fuel consumption per year for the same type of work from locomotive class 52 is:

$$Dieselfuel = (240 workingdays)(80\,L) = 19200\,L$$

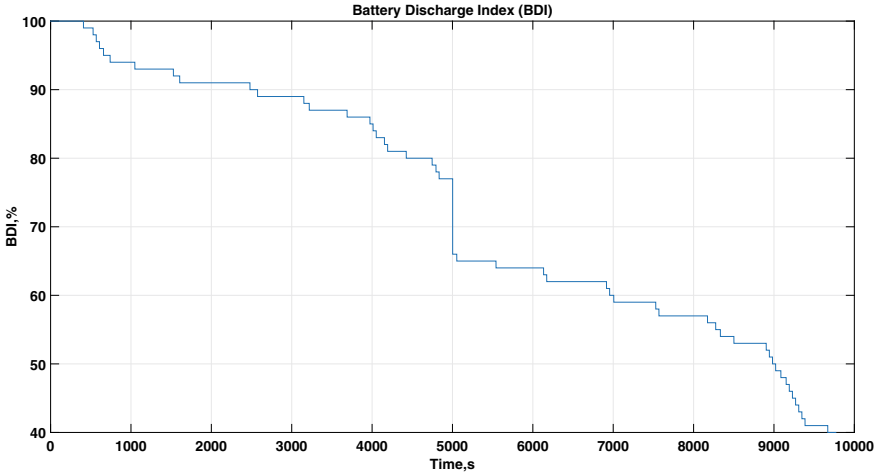


Fig. 8 Battery discharge index for Lead-Acid

Financially, the use of battery locomotives over conventional shunting diesel locomotives makes it possible to lower the total cost of maneuvering where possible. This important aspect of savings would push more and more companies to invest in this type of technology.

Consumption of energy from the battery illustrates Fig. 8. It could be seen that for the given work cycle, loading and shunting of 12 chamotte wagons, the battery has rendered all of its energy up to 40.

For larger volumes of work, the capacity of the battery can be increased in accordance with a particular application. The locomotive works during different periods of the year under different climatic conditions in a moderate climate with success and without failures or change in performance. At ambient temperatures of −20C, changes in battery capacity do not have a significant effect on the performance. Despite the small number of Lead-Acid Battery cycles and shorter lifetime than alternative options, their life may exceed 10 years in normal exploitation. At this price point these type of batteries used in vehicles remain the cheapest solution [9, 20, 28].

Thanks to the technological progress in battery locomotives and the possibilities for regenerative braking, there is a tendency towards more and more mass use of this type of locomotives before the conventional diesel locomotives. New environmental standards and high maintenance costs make diesel engines unattractive for such applications. Battery locomotives also solve another major problem facing shunting locomotives, namely the high percentage of Idle in their use. From 2018 in Bulgaria there is a process of replacement of locomotives Class 52 and Class 55 in industrial plant and their replacement with modern battery locomotives. Through this step, many industrial plants seek to reduce their footprint on nature while reducing their costs in response to market needs. In the next decades, battery locomotives will

occupy an increasing share of shunting work and there is an intensive replacement of old diesel locomotives with battery ones.

As a disadvantage of regenerative technologies for the moment, the fact of a lack of emergency braking capacity can be highlighted. Now existing systems can not cope optimally with the task of quickly absorbing the energy from an emergency train stop. For these cases, an additional dynamic brake braking mechanism or conventional pneumatic brakes is required.

5 Conclusion

This article explores the energy efficiency optimization of regenerative braking in shunting locomotives instead of the conventional braking with the automatic train brake. Lead-Acid technology has been studied to power and store energy. Regenerative braking optimizes shunting costs and reduces wear on train braking systems. Electric battery shunting locomotives provide a real opportunity to reduce emissions in cities and industrial plants. The reduction of harmful emissions in railway transport has emerged as a strategic task for the Union of Railways with a database of technologies enabling optimization of the energy efficiency. The ability to charge the battery from renewable energy sources allows us to achieve zero emission. In the future work some other techniques for energy optimization will be discussed.

Acknowledgements Venelin Todorov is supported by the Bulgarian National Science Fund under Project KP-06-M32/2-17.12.2019 "Advanced Stochastic and Deterministic Approaches for Large-Scale Problems of Computational Mathematics" and by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security" (ICTinSES), contract No. D01-205/23.11.2018, financed by the Ministry of Education and Science. The work is also supported by the Bulgarian National Fund of Science under Projects DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large-Scale Computational Problems" and by Project KP-06-Russia/17 "New Highly Efficient Stochastic Simulation Methods and Applications".

References

1. http://www.railway-research.org/IMG/pdf/r.2.2.3.1.pdf (Accessed on 1 May 2019)
2. http://lz1866.com/battery-locomotives#es (Accessed on 2 May 2019)
3. Ogasa, M.: Energy saving and environmental measures in railway technologies: example with hybrid electric railway vehicles. IEEJ Trans. Electr. Electron. Eng. **3**(1), 15–20 (2008). https://doi.org/10.1002/tee.20227
4. Khodaparastan, M., Mohamed, A.A., Brandauer, W.: Recuperation of regenerative braking energy in electric rail transit systems. IEEE Trans. Intell. Transp. Syst. **20**(8), 2831–2847 (2019). https://doi.org/10.1109/TITS.2018.2886809
5. Takagi, R.: Energy saving techniques for the power feeding network of electric railways. IEEJ Trans. Electr. Electron. Eng. **5**(3), 312–316 (2010). https://doi.org/10.1002/tee.20535

6. Shimada, M., Oishi, R., Araki, D., Nakamura, Y.: Energy storage system for effective use of regenerative energy in electrified railways. Hitachi Rev. **59**(1), 33–38 (2010). ISSN 0018277X
7. Saleh, M., Dutta, O., Esa, Y., Mohamed, A.: Quantitative analysis of regenerative energy in electric rail traction systems. 2017 IEEE Ind. Appl. Soc. Annu. Meet. IAS 2017, **2017**, 1–7 (2017). R. Nicole, Title of paper with only first word capitalized, J. Name Stand. Abbrev (in press). https://doi.org/10.1109/IAS.2017.8101774
8. https://www.alstom.com/sites/alstom.com/files/2018/10/30/re-use_laymans_report_en.pdf (Accessed on 9 May 2019)
9. UIC, Energy Efficiency Technologies for Railways. http://www.railway-energy.org (Accessed on 9 May 2019)
10. Steiner, M., Scholten, J.: Energy storage on board of railway vehicles, 2005 Eur. Conf. Power Electron. Appl. **2005**, 1–10 (2005). https://doi.org/10.1002/tee.20535
11. Takagi, R.: Application of energy storage systems for DC electric railways. WIT Trans. Ecol. Environ. **121**, 527–535 (2009). https://doi.org/10.2495/ESU090481
12. Luo, X., Wang, J., Dooner, M., Clarke, J.: Overview of current development in electrical energy storage technologies and the application potential in power system operation. Appl. Energy **137**, 511–536 (2015). https://doi.org/10.1016/j.apenergy.2014.09.081
13. Gonzalez-Gil, A., Palacin, R., Batty, P.: Sustainable urban rail systems: Strategies and technologies for optimal management of regenerative braking energy. Energy Convers. Manag. **75**, 374–388 (2013). https://doi.org/10.1016/j.enconman.2013.06.039
14. Gunselmann, W.: Technologies for increased energy efficiency in railway systems, 2005 Eur. Conf. Power Electron. Appl. **2005**, 1–10 (2005). https://doi.org/10.1109/epe.2005.219712
15. Hu, J., Zhao, Y., Liu, X.: The design of regeneration braking system in light rail vehicle using energy-storage Ultra-capacitor, 2008 IEEE Veh. Power Propuls. Conf. VPPC **2008**, 1–5 (2008). https://doi.org/10.1109/VPPC.2008.4677708
16. Miyatake, M., Matsuda, K.: Energy saving speed and charge/discharge control of a railway vehicle with on-board energy storage by means of an optimization model. IEEJ Trans. Electr. Electron. Eng. **4**(6), 771–778 (2009). https://doi.org/10.1002/tee.20479
17. European Commission, Electrification of the Transport System: Studies and Reports, pp. 1–49 (2017). ISBN:9789279718564
18. Fernandez-Rodriguez, A. et al.: Charging electric vehicles using regenerated energy from urban railways. In: 2017 IEEE Vechile Power Propuls. Conference VPPC 2017 - Proceedings, vol. 2018, pp. 1–6 (2018). https://doi.org/10.1109/VPPC.2017.8330998
19. Ogura, K., et al.: Test results of a high capacity wayside energy storage system using Ni-MH batteries for DC electric railway at New York City Transit, 2011 IEEE Green Technol. Conf. Green **2011**, 6–11 (2011). https://doi.org/10.1109/GREEN.2011.5754859
20. Ceraolo, M., Lutzemberger, G.: Stationary and on-board storage systems to enhance energy and cost efficiency of tramways. J. Power Sources **264**, 128–139 (2014). https://doi.org/10.1016/j.jpowsour.2014.04.070
21. Iannuzzi, D., Tricoli, P.: Speed-based state-of-charge tracking control for metro trains with onboard supercapacitors. IEEE Trans. Power Electron. **27**(4), 2129–2140 (2012). https://doi.org/10.1109/TPEL.2011.2167633
22. Iannuzzi, D., Tricoli, P.: Supercapacitor state of charge control based on changeover finite state controller for metro-train applications. In: 3rd International Confernce Clean Electric Power Renew. Energy Resources Impact, ICCEP 2011, pp. 550–556 (2011)
23. Mir, L., Etxeberria-Otadui, I., De Arenaza, I.P., Sarasola, I., Nieva, T.: A supercapacitor based light rail vehicle: System design and operations modes. In: 2009 IEEE Energy Conversion Congress Expo. ECCE 2009, vol. 146, pp. 1632–1639 (2009). https://doi.org/10.1109/ECCE.2009.5316073
24. Steiner, M., Scholten, J.: Energy storage on board of railway vehicles, 2005 Eur. Conf. Power Electron. Appl. **2005**, 1–10 (2005). https://doi.org/10.1109/epe.2005.219410
25. Grimaud, D., Levraut, J.: Optimal driving strategy for traction energy saving on DC suburban railways. Minerva Anestesiol. **67**(4), 263–270 (2001). https://doi.org/10.1049/iet-epa

26. Gotsov, T., Todorov, V., Kolev, T. (2020). Energetic Optimization of the Use of Battery Shunting Locomotive in Industrial Plant with Regenerative Brake. Communication Papers of the 2020 Federated Conference on Computer Science and Information Systems, 23, 3–6. https://doi.org/10.15439/2020f117
27. Frilli, A., Meli, E., Nocciolini, D., Pugi, L., Rindi, A.: Energetic optimization of regenerative braking for high speed railway systems. Energy Convers. Manag. **129**, (2016). https://doi.org/10.1016/j.enconman.2016.10.011
28. Shtang, A.A., Yaroslavtsev, M.V.: Battery-electric shunting locomotive with lithium-polymer storage batteries. Proceedings - 2016 11th International Forum on Strategic Technology. IFOST **2016**, 162–165 (2017). https://doi.org/10.1109/IFOST.2016.7884218
29. Hillmansen, S., Roberts, C.: Energy storage devices in hybrid railway vehicles: A kinematic analysis. Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit **221**(1), 135–143 (2007). https://doi.org/10.1243/09544097JRRT99

Author Index

A
Aïder, Méziane
Alihodzic, Adis
Amorim, Pedro
Andonov, Velin
Apostolov, Stoyan

B
Baatout, Fatma Zohra
Bendali, Fatiha

D
Dezert, Jean
Dimitrov, Yuri
Dimov, Ivan
Dörpinghaus, Jens

F
Fadda, Edoardo
Fidanova, Stefka

G
Ganzha, Maria
Georgieva, Rayna
Gotsov, Tsvetomir
Goullieux, Adeline
Grigoreva, Natalia

H
Habarta, Filip
Hifi, Mhand

K
Kirilov, Leoneed

L
Li, Yuanyuan

M
Mailfert, Jean
Malá, Ivana
Manerba, Daniele
Marek, Luboš
Matrenin, Pavel
Mikhov, Rossen
Miryanov, Radan
Miyazawa, Flávio K.
Mole Kamga, Eloise
Mucherino, Antonio
Myasnichenko, Vladimir

O
Ostromsky, Tzvetan

P
Poryazov, Stoyan

Q
Quilliot, Alain

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

S. Fidanova (ed.), *Recent Advances in Computational Optimization*, Studies in Computational Intelligence 986,
https://doi.org/10.1007/978-3-030-82397-9

R
Rios, Brenner H. O.
Roeva, Olympia
Roohnavazfar, Mina

S
Sadeghsa, Shohre
Saranova, Emiliya
Sdobnyakov, Nickolay
Soheil Hosseini, Seyed
Sokolov, Denis
Stefan, Andreas
Štěpánek, Lubomír

T
Tadei, Roberto
Tchamova, Albena
Terzo, Olivier
Tian, Tianhai
Todorov, Daniel
Todorov, Venelin
Toussaint, Helene
Traneva, Velichka
Tranev, Stoyan

W
Wormald, Nick

X
Xavier, Eduardo C.

Z
Zlatev, Zahari