IntechOpen

Optimisation Algorithms and Swarm Intelligence

Edited by Nodari Vakhania
and Mehmet Emin Aydin

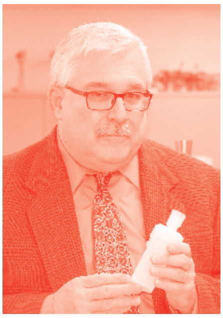
Optimisation Algorithms and Swarm Intelligence

Edited by Nodari Vakhania and Mehmet Emin Aydin

Published in London, United Kingdom

IntechOpen

Supporting open minds since 2005



Optimisation Algorithms and Swarm Intelligence
http://dx.doi.org/10.5772/intechopen.94617
Edited by Nodari Vakhania and Mehmet Emin Aydin

Contributors
Ersin Büyük, Preethi Gunishetty Devarakonda, Bojan Bozic, Clement N. Nyirenda, Nikolaos L. Tsitsas, Alkmini Michaloglou, Zsolt T. Kosztyan, Leila Gharsalli, Mehmet Emin Aydin, Rafet Durgut

© The Editor(s) and the Author(s) 2022
The rights of the editor(s) and the author(s) have been asserted in accordance with the Copyright, Designs and Patents Act 1988. All rights to the book as a whole are reserved by INTECHOPEN LIMITED. The book as a whole (compilation) cannot be reproduced, distributed or used for commercial or non-commercial purposes without INTECHOPEN LIMITED’s written permission. Enquiries concerning the use of the book should be directed to INTECHOPEN LIMITED rights and permissions department (permissions@intechopen.com).
Violations are liable to prosecution under the governing Copyright Law.

CC BY

Individual chapters of this publication are distributed under the terms of the Creative Commons Attribution 3.0 Unported License which permits commercial use, distribution and reproduction of the individual chapters, provided the original author(s) and source publication are appropriately acknowledged. If so indicated, certain images may not be included under the Creative Commons license. In such cases users will need to obtain permission from the license holder to reproduce the material. More details and guidelines concerning content reuse and adaptation can be found at http://www.intechopen.com/copyright-policy.html.

Notice
Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

First published in London, United Kingdom, 2022 by IntechOpen
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, 5 Princes Gate Court, London, SW7 2QJ, United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Optimisation Algorithms and Swarm Intelligence
Edited by Nodari Vakhania and Mehmet Emin Aydin
p. cm.
Print ISBN 978-1-83968-665-8
Online ISBN 978-1-83968-666-5
eBook (PDF) ISBN 978-1-83968-667-2

We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,100+
Open access books available

149,000+
International authors and editors

185M+
Downloads

156
Countries delivered to

Our authors are among the
Top 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index (BKCI)
in Web of Science Core Collection™

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Meet the editors

Nodari Vakhania graduated with honors from the Faculty of Applied Mathematics and Cybernetics, Tbilisi State University, Georgia, in 1983. In 1989, he entered the Ph.D. program in Computer Science at the Russian Academy of Sciences, Moscow, and obtained a degree in Mathematical Cybernetics in 1991. In 1992, he was a postdoctoral fellow at the Russian Academy of Sciences and had a short-term visiting position at the University of Saarbrucken, Germany. In 1995, he became a professor at the Centro de Investigación en Ciencias, State University of Morelos, Mexico. He also has an honorary position at the Institute of Computational Mathematics of the Georgian Academy of Sciences, where he received a doctoral degree in Mathematical Cybernetics in 2004. His research interests include the design and analysis of algorithms, discrete optimization, computational complexity, and scheduling theory. He is an author of nearly 100 refereed research papers including more than 60 publications in highly ranked international journals. He has also worked with different scientific committees, including those at the Mexican Science Foundation CONACyT. He is an editorial board member and a referee for several international scientific journals. He has obtained research grants and honors in Germany, France, the Netherlands, United States, Russia, and Mexico, and has given more than forty invited talks throughout the world.

Mehmet Emin Aydin is a Senior Lecturer in Computer Science at, the University of the West of England. He obtained a BSc, MA, and Ph.D. from Istanbul Technical University, Istanbul University, and Sakarya University, Turkey, respectively. His research interests include machine learning, intelligent agents and multi-agent systems, metaheuristics, swarm intelligence, resource planning, scheduling and optimization, and combinatorial optimization. He has served as guest editor for special issues of peer-reviewed international journals. In addition to being a member of the advisory committees of many international conferences, Dr. Aydin is an editorial board member for various journals. He is currently a fellow of the Higher Education Academy, a member of Engineering and Physical Sciences Research Council (EPSRC) College, UK, and a senior member of the Association of Computing Machinery (ACM) and Institute of Electrical and Electronics Engineers (IEEE).

Contents

Preface

Computational algorithms nowadays play a definitive role is most real-life applications, from mobile phones to supercomputers, Internet servers, manufacturing, and more. A computational algorithm is designed to optimize some measure for a given problem. A discrete algorithm built for an optimization problem normally deals with a large set of feasible solutions (ones that respect problem restrictions) and tries to find a feasible solution that fits better-given optimization criteria. Hence, an intelligent method for the enumeration of feasible solutions may lead to an efficient computational algorithm. This book contains some recent work on optimization methods and heuristics for complex real-life problems that aim to find suitable and feasible solutions. For example, it includes information on Particle Swarm Optimization (PSO), which is one of the renowned swarm optimization approaches. It has been used over the last few decades in various forms and capacities. Due to its sound foundation, it offers many stable and substantial solutions for complex and difficult real-world problems. Recently, it has been used in more modern and emerging complex problems such as training of deep neural networks alongside design and resource management problems of modern wireless communication networks. The book also provides useful guidance on the implementation and use of PSO in various problems.

Chapter 1 presents novel improvement strategies for a recently developed swarm intelligence algorithm called the Crow Search Algorithm (CSA). This type of swarm intelligence algorithm is motivated by the social behavior of crow flocks. In a CSA, given a current solution, its neighbor, to be considered next, is determined randomly. This may lead to the convergence to a local optimum, due to the local search that is typically used in such algorithms. The chapter considers the multi-strategy search that uses different search strategies for the selection of each next solution. This augmented version of CSA performs well in practice.

Chapter 2 considers a hybrid genetic approach for the solution of complex optimization problems. It reviews different types of genetic algorithms and considers the existing methods of integration of some search and optimization techniques within the genetic algorithm framework. As an illustrative sample, the chapter proposes a new hybrid genetic algorithm for the mechanical sizing of a composite structure located in the upper part of a launcher. Given structural constraints that define the feasible solution space, the objective here is to minimize the overall mass of the construction.

Chapter 3 deals with project management issues, concentrating on scheduling strategies for optimal project management. It reviews related methods and algorithms for project management and suggests a matrix-based model and a scheduling algorithm that can handle renewable resources. The framework can also be extended to handle nonrenewable resources. The proposed risk evaluation tool is used to compare the efficiency of different scheduling rules for project management.

Chapter 4 presents an approach to help train convolutional neural networks in the recognition of human activities. Recently, search algorithms, especially swarm intelligence algorithms, have been utilized in training machine learning models.

Chapter 5 introduces another PSO algorithm to solve the wave scattering problem in inhomogeneous media. Variants of PSO enhanced with chaos theory and other accelerating components have been implemented and tested for better performance. The chapter describes and discusses the results of this testing.

Chapter 6 investigates the efficiency issues surrounding the use of PSO in the gateway placement problem of short- and long-range networks designed for Internet of Things (IoT) implementations. One of these networks is LoRaWAN, which requires gateway placement for high efficiency. Different variants of PSO have been developed and tested to tackle efficiency issues. This chapter reviews these algorithms and proposes the best-performing PSO for the problem.

Chapter 7 discusses PSO and its multi-objective implementations. Research on multi-objective optimization has increased due to it being more realistic for solving real-world problems. PSO has been used in multi-objective studies for solving geophysical problems and has demonstrated good success, as described in this chapter.

Nodari Vakhania
Professor,
Centro de Investigación en Ciencias,
Universidad Autónoma del Estado de Morelos,
Cuernavaca, Mexico

Mehmet Emin Aydin
University of the West of England,
Bristol, England

Chapter 1

Multi Strategy Search with Crow Search Algorithm

Rafet Durgut and Mehmet Emin Aydin

Abstract

Crow Search Algorithm (CSA) is one of the recently proposed swarm intelligence algorithms developed inspiring of the social behaviour of crow flocks. One of the drawbacks of the original CSA is that it tends to randomly select a neighbour on search strategy due to its low convergence rate, which pushes the search to stick in local optima due to the same search strategy applied across iterations. The multi-strategy search for CSA (CSA-MSS) has been proposed to enrich the search facilities and provide diversity to overcome these drawbacks. The multi-strategy search implies utilising a pool of strategies consists of six different types of search operators. The multi-strategy approach with a selection mechanism has not been proposed for CSA before and implemented first time. The comparative performance analysis for the proposed algorithm has been conducted over solving 24 benchmark problems. The results demonstrate that the proposed approach is outperforming well-known state-of-the-art methods.

Keywords: swarm intelligence, crow search algorithm, global optimisation, strategy selection, operator selection schemes

1. Introduction

Over the last two decades, many nature-inspired metaheuristic optimisation algorithms (MOAs) have been proposed to solve many abstract and real-world problems including combinatorial, binary and real engineering problems [1]. MOAs are generally iterative processes developed imitating natural phenomena and animal behaviours to obtain optimal solutions for given problems [2]. Although these algorithms provide satisfactory solutions to various optimisation problems, they do not always guarantee an optimal way of solving them in different scales [3, 4]. In this respect, MOAs has recently received increasing interest from scholars in related fields (e.g. engineering, business, logistics, etc.) as a way to design algorithms that specify a solution to general problems using a certain pattern of behaviour inspired by a given society. An example in this respect is the study of a colony of insects or the behaviour of other animals' societies [5]. Swarm intelligence algorithms tend to simulate social behaviours of the imitated societies, where some renown swarm intelligence algorithms are particle swarm optimisation (PSO) inspired of bird flocks and fish schooling [6, 7], artificial bee colony (ABC) simulates collective food searching behaviour of honey bees [8, 9], cuckoo search algorithm is based on the brood parasitism of some cuckoo species [10, 11], group search optimiser (GSO) is based on animal searching behaviour [12, 13], and firefly algorithm (FA) is based on the flashing light patterns of tropic fireflies [14, 15].

Crow Search Algorithm (CSA) is another swarm intelligence metaheuristic framework inspired of feeding behaviours of crow flocks [1]. Similar to many other MOAs, CSA needs a balance in between exploration and exploitation activities throughout the search process in order to approximate global optimal solutions [16]. In this respect, exploitation refers to the use of gained experience in deepening the search within the locality of the current solution while exploration implies diversification, e.g. jumping to another region of the search space, in order to reach the optimal solution. The majority of the real-world problems are multi-model problems, which possess numerous local optimum solutions hard to avoid, the exploitative search has higher probability to get stuck into a local optimum. In contrast, exploration helps enrich the search with higher diversity to let the search move to other feasible regions within the search space, which may end up with a better payoff due to that a new region may lead to global best. In addition, a well-balanced exploration would let the algorithm to avoid sub-optimal solutions. It is known that a search strategy with balance in between exploration and exploitation facilitates finding sub-optimal solutions close to the global optimal [17] in shorter time, while CSA is known with its higher exploitation capability as designed [18].

In order to balance exploration and exploitation capabilities for MOAs, a number of approaches have been implemented; among these, three categories can be highlighted; (i) dynamic parameterisation on the fly [19], (ii) implementing different perturbation search strategies [20] and (iii) using multiple strategies in the search [21] instead of single one. For the latest approach, a strategy pool consisting of different strategies is constructed. The algorithm chooses the appropriate strategy according to search process circumstances and dynamics. In this article, an implementation of CSA approach with multiple strategies is introduced to solve functional optimisation problems using a pool of six different search strategies. The main contribution of the study is the use of multi-strategy search policy with CSA and suggesting a selection scheme in order to respond best to the search dynamics for solving functional optimisation problems.

2. Related works

Studies related to this work can be viewed into two categories: modified version of CSA and adaptation approaches. Zamani et al. [22] proposed new version of CSA, called Conscious Neighbourhood-based CSA (CCSA), with three new search strategies to improve local and global search capabilities of CSA. Sonuc [23] proposed BinCSA, a pure binary version of CSA, in which a logic gate was used as a search strategy. Majhi et al. [24] developed a new version of CSA, called OBL-CSA-MO, in which they tried to escape from local optima by using opposition learning strategy and mutation operator.

CSA has two control parameters: namely awareness probability (AP) and flight length (FL). AP is randomly generated solution probability and controls intensification and diversification balance. FL controls searching effect of algorithm. Rizk-Allah et al. [25] proposed a new version of CSA integrated with chaos theory for solving fractional optimisation problems. Coelho et al. [26] developed a modified CSA in which Gaussian distribution function is used in order to control two algorithmic parameters. Necira et al. [27] proposed an enhanced version of CSA, called dynamic crow search algorithm (DCSA), in which there are two modifications on CSA. Awareness probability is linearly adjusted throughout iterations, and FL is expressed as pareto probability density function. There are several studies aiming to improve the efficiency and robustness of CSA; for more details and further modifications of CSA, the reader is referred to the surveys [28, 29]. To the best knowledge of the authors, CSA has not been studied with multiple search strategies to avoid local optima problem, and this article presents the first ever study in this respect.

3. Materials and methods

3.1 Crow search algorithm

Crows are known to be among the smartest bird species based on their natural talents [30]; they can remember the location where they or other bird species store food [31], often times even called 'thieves'. With such a sound memory skill, they can even recall faces and warn the flock using complex communication channels in dangerous situations [32]. CSA is developed as a metaheuristic optimisation algorithm inspired by such smart behaviours [1] where each crow represents a valid solution within the search space. The crow's memory represents the best solutions for the given problem set. Let N be the number of crows in the flock and the search space be D dimensional. Then the position of the ith crow at jth time should be as $x^{i,j} = x_d^{i,j}$, where $d \in D$, $i \in N$ and $j \in T$.

The position of the information held by each crow in the memory should be $m^{i,j}$, where m information stored in memory holds the best position information available up to that iteration, and T is the maximum iteration number. There are two options once the crows move towards the food positions in mind: (i) moving towards position m blind of being followed by another crow (k); (ii) moving to any random position aware of being followed by another curious crow. The choice between these two options is modelled as follows:

$$x^{i,j+1} = \begin{cases} x^{i,j} + r_i \times FL \times \left(m^{k,j} - x^{i,j}\right), & r_j > AP \\ \text{a random position}, & otherwise \end{cases} \tag{1}$$

where r_i and r_j are two uniformly distributed random variables within the range of $[0, 1]$, FL is the flight distance, and AP is the probability of awareness, k is a randomly specified integer parameter that takes value from 1 to N. If the obtained position information of the solution—new solution—is better than the one in the memory, then, the position information is updated. The exploration and exploitation capabilities are managed with FL and AP in this approach, where FL is used to obtain better solutions, while AP is used to discover new solutions. If a high value is set for the AP, it will then come into play at the local minimum in the exploration phase of the algorithm. If a low value is set for the AP, late convergence will occur in the exploitation phase. Therefore, the AP value needs to be adjusted well [1].

The general overview of the CSA is given in Algorithm 1 in a pseudo code. Firstly, the algorithmic parameters are determined based on the position of each crow randomly generated in a d-dimensional search space. Once the first population is adopted as the initial best set of solution in the flock memory. Then the main loop of the algorithm starts working. Each crow in the flock makes decision for whom to follow within the flock. Next, the probability of AP is calculated randomly, as a threshold value, to identify if the crow is to move towards the food or to reach a new random position. If the new position information is a valid/feasible one and better than the solution in the memory, the crow updates its own memory. The method continues until the specified criteria are obtained.

Algorithm 1. The pseudo code of the CSA.

```
Initialising of parameters ($D$, $T$, $AP$, $FL$)
Randomly initialise crows positions
Memorise the positions
**while** termination criteria is not met **do**
    **for** i = 1:N **do**
        Selection Phase
        Determine AP
        **if** rand() > = AP **then**
            Calculate $x^{i,j+1}$ by the Eq. (1)
        **else**
            Generate Random Position
        **end if**
    **end for**
    Bound Control
    Evaluate
    Memorise
**end while**
```

3.2 Search strategies

Search strategies are used to help algorithms to switch from one solution to another within the neighbourhood. However, use of single search strategy with any metaheuristic algorithm ends up with serious limitations with respect to the balance in between the exploration ant exploitation capabilities. In order to deliver search with a balance, we decided to use a pool of search strategies composed of six different search strategies. The five different approaches picked up from to the literature inspired of the update rules used in differential evolution (DE) and artificial bee colony (ABC) algorithms, where some modifications applied [5, 33].

Eq. (2) is the original update rule used by CSA to generate candidate solution, while Eq. (3) is extended with global best guided search strategy (GCSA) [34] in order to improve exploitation capability. Eq. (4) is a global best-guided search strategy embedded with flock memory (BCSA). Eq. (5) is added to the strategy pool to enhance exploration ability with which it brings two different randomly chosen crows' position in the scene (CBCSA). Eq. (6) is developed inspiring of DE/RAND/1 to offer higher exploration capability (RCSA). Eq. (7) is also an implementation of DE/RAND/1 rule inserting memory solutions in use (RMCSA). In the all strategies, *k* is a randomly selected neighbour solution from the flock.

$$x^{i,j+1} = x^{i,j} + r_i \times FL \times \left(m^{i,j} - x^{i,j}\right) \tag{2}$$

$$x^{i,j+1} = x^{i,j} + r_i \times FL \times \left\{\left(m^{k,j} - x^{i,j}\right) + \left(g_{best} - x^{i,j}\right)\right\} \tag{3}$$

$$x^{i,j+1} = g_{best} + r_i \times FL \times \left(m^{k_1,j} - m^{k_2,j}\right) \tag{4}$$

$$x^{i,j+1} = x^{i,j} + r_i \times \left\{\left(m^{i,j} - x^{i,j}\right) + \left(x^{k_1,j} - x^{k_2,j}\right)\right\} \tag{5}$$

$$x^{i,j+1} = x^{k_1,j} + r_i \times \left(x^{k_2,j} - x^{k_3,j}\right) \tag{6}$$

$$x^{i,j+1} = m^{k_1,j} + r_i \times \left(m^{k_2,j} - m^{k_3,j}\right) \tag{7}$$

where $i = 1, .., N$ and $j = 1, .., T$ are indexes for the number of crows and iterations, respectively, while k_1, k_2, k_3 are three random integers required to hold $k_1 \neq k_2 \neq k_3$.

3.3 CSA with multi-strategy search (CSA-MSS)

As explained in the previous section, search-based on single strategy imposes limitations which enforce local optima. This renowned problem is tackled with various ways in the literature. Memetic algorithms [35], modified swarm intelligence algorithms [36], hybrid solutions [37] are the studies seeking for resolutions for this issue. Recently, adaptive operator selection schemes [8, 38] look promising to overcome this issue in an easier way. The proposed approach in this study is based on a multi-strategy search approach, CSA-MSS, which implies adaptive use of multiple search strategies in order to overcome the limitations of single strategy search. It consists of two main components: credit assignment and strategy selection. In order to evaluate the performance of the applied strategy, reward value, $r_{i,j}$, is calculated as follows:

$$r_{i,j} = \frac{f(x)}{g_{best}}\left(f(x) - f(x')\right) \tag{8}$$

where$f(x)$ is the objective value of the current solution, and$f(x')$ is the objective value of the candidate solution which is generated by search strategy i. To balance reward value in the course of iterations, improvement on objective function is normalised over global best solution value. In Eq. (9), credit values are assigned to each strategy.

$$c_{i,j+1} = (1 - \alpha)c_{i,j} + \alpha\left(\frac{r_{i,j}}{s_{i,j} + 1}\right) \tag{9}$$

where α is the learning rate, $c_{i,j}$ and $s_{i,j}$ are credit value and the number of successful improvements so far for strategy i at iteration j, respectively. The average reward is calculated for iteration j.

Adaptive pursuit approach is used as the strategy selection scheme for selecting the search strategy to operate due to its proven good performance in literature [8]. In this selection scheme, each strategy has a probability value which is calculated with Eq. (10).

$$p_{i,j+1} = \begin{cases} p_{i,j} + \beta\left(p_{max} - p_{i,j}\right), & \text{if } i = i_j^* \\ p_{i,j} + \beta\left(p_{min} - p_{i,j}\right), & \text{otherwise} \end{cases} \tag{10}$$

where β is the learning rate in $[0, 1]$, p_{min} is the minimum selection probability, and i_j* is the best strategy for iteration i. In this approach, the best strategy wins the maximum selection probability. The general overview of the CSA-MSS is given in Algorithm 2. The proposed approach starts with selecting a strategy to operate. If there is any non-rewarded strategy within the pool, a random selection is conducted to let these strategies have a chance to be opted. The selected strategy is operated on the current solution to generate a candidate solution—a neighbouring solution. A reward value is calculated for the opted strategy and recorded if it is successful in the running iteration. The credit values are updated from the reward values obtained/recorded at the end of each iteration. Both average and maximum reward values are calculated through a sliding window over the last W iterations.

Algorithm 2. The general overview of CSA-MSS.

```
Initialising of parameters $(D,T_{max},M,AP,FL)$
Randomly initialise crows positions
Memorise the positions
while termination criteria is not met do
    for i = 1:N do
        Selection Phase
        Determine AP
        if rand() $>=$ AP then
            if Any strategy has not rewarded then
                Select the strategy randomly
            else
                Assign probabilities using Eq. (10)
                Select the strategy using roulette-wheel selection
            end if
            Generate candidate solution
        else
            Generate Random Position
        end if
        Bound Control
        Evaluate
        Calculate reward using Eq. (8)
        if reward $>$ 0 then
            Add reward and increase s for selected strategy
        end if
    end for
    Memorise
    Credit assignment using Eq. (9)
end while
```

3.4 Benchmark functions

In order to compare and analyse the performance of the approaches under-consideration alongside CSA, a number of well-known numeric benchmark functions have been used, which are collected from the literature [39, 40] as in **Table 1**. The characteristics of each are depicted in the second column with UN, MN, SP and NP standing for unimodality, multimodality, separable and non-separable, respectively. If the functions have only one local optima as the global optima, then it is called as unimodal function [41]. These types of functions can be used to demonstrate exploitation ability of the algorithms. If the functions have more than one local optima, then it is called a multimodal function, where exploration is also needed significantly to travel among the regions, i.e. modals, that have their local optima. If the relationship between any two inputs of a function is loose, i.e. can be easily operated standalone, then it is considered separable, non-separable otherwise [42].

No	Name	Search Space	C	Function
F1	Sphere	$[-100, 100]^D$	US	$f_1\left(\vec{x}\right) = \sum_{i=1}^{n} x_i^2$
F2	Elliptic	$[-100, 100]^D$	UN	$f_2\left(\vec{x}\right) = \sum_{i=1}^{n} x_i^2 \left(10^6\right)^{(i-1)/(n-1)}$
F3	Sum Squares	$[-10, 10]^D$	US	$f_3\left(\vec{x}\right) = \sum_{i=1}^{n} i x_i^2$
F4	Sum Power	$[-10, 10]^D$	MS	$f_4\left(\vec{x}\right) = \sum_{i=1}^{n} \lvert x_i \rvert^{(i+1)}$
F5	Schwefel 2.22	$[-10, 10]^D$	UN	$f_5\left(\vec{x}\right) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
F6	Schwefel 2.21	$[-100, 100]^D$	UN	$f_6\left(\vec{x}\right) = \max_i \{\lvert x_i \rvert,\ 1 \le i \le n\}$
F7	Step	$[-100, 100]^D$	US	$f_7\left(\vec{x}\right) = \sum_{i=1}^{n} \left(\lfloor x_i + 0.5 \rfloor\right)^2$
F8	Quartic	$[-1.28, 1.28]^D$	US	$f_8\left(\vec{x}\right) = \sum_{i=1}^{n} i x_i^4$
F9	QuarticWN	$[-1.28, 1.28]^D$	US	$f_9\left(\vec{x}\right) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$
F10	Rosenbrock	$[-10, 10]^D$	UN	$f_{10}\left(\vec{x}\right) = \sum_{i=1}^{n} \left[100\left(x_{i+1} - x_i^2\right) + \left(x_i - 1^2\right)\right]$
F11	Rastrigin	$[-5.12, 5.12]^D$	MS	$f_{11}\left(\vec{x}\right) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right]$
F12	Non-Continuous Rastrigin	$[-5.12, 5.12]^D$	MS	$f_{12}\left(\vec{x}\right) = \sum_{i=1}^{n} \left[y_i^2 - 10\cos\left(2\pi y_i\right) + 10\right]\ y_i = \begin{cases} x_i, & \lvert x_i \rvert < \frac{1}{2} \\ \frac{round(2x_i)}{2}, & \lvert x_i \rvert \ge \frac{1}{2} \end{cases}$
F13	Griewank	$[-600, 100]^D$	MN	$f_{13}\left(\vec{x}\right) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
F14	Schwefel 2.26	$[-500, 500]^D$	UN	$f_{14}\left(\vec{x}\right) = 418.98 * n \sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$
F15	Ackley	$[-32, 32]^D$	MN	$f_{15}\left(\vec{x}\right) = -20\, exp\left\{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right\} - exp\left\{\frac{1}{n}\sum_{i=1}^{n} \cos\left(2\pi x_i\right\} + 20 + e$

No	Name	Search Space	C	Function
F16	Penalized1	$[-50, 50]^D$	MN	$f_{16}\left(\vec{x}\right) = \frac{\pi}{n}\left\{10\sin^2(\pi x_1) + \sum_{i=1}^{n-1}(\mathrm{y_i} - 1)^2\ \left[1 + 10\sin^2(\pi y_{i+1})\right] + \left(y_n - 1\right)^2\right\} + \sum_{i=1}^{n} u(x_i, 10,\ 100, 4)\, y_i =$ $1 + \frac{1}{4}(x_i + 1) \quad u_{x_i,a,k,m} = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(x_i - a)^m, & x_i < -a \end{cases}$
F17	Penalized2	$[-50, 50]^D$	MN	$f_{17}\left(\vec{x}\right) = \frac{1}{10}\left\{\sin^2(\pi x_1) + \sum_{i=1}^{n-1}(\mathrm{x_i} - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_{i+1})\right]\right\} + \sum_{i=1}^{n} u(x_i, 5,\ 100, 4)$
F18	Alpine	$[-10, 10]^D$	MS	$f_{18}\left(\vec{x}\right) = \sum_{i=1}^{n}\lvert x_i \sin(x_i) + 0.1x_i\rvert$
F19	Levy	$[-10, 10]^D$	MN	$f_{19}\left(\vec{x}\right) = \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(\pi x_{i+1})\right] + \sin^2(3\pi x_{i+1}) + \lvert x_n - 1\rvert\left[1 + \sin^2(3\pi x_n)\right]$
F20	Weierstrass	$[-0.5, 0.5]^D$	MN	$f_{20}\left(\vec{x}\right) = \sum_{i=1}^{n}\left(\sum_{k=0}^{k_{max}}\left[a^k\cos\left(2\pi b^k(x_i + 0.5)\right)\right]\right) - D\sum_{k=0}^{k_{max}} a^k\, cos\left(2\pi b^k 0.5\right) \quad a = 0.5, \quad b = 3, \quad k_{max} = 20$
F21	Schaffer	$[-100, 100]^D$	MN	$f_{21}\left(\vec{x}\right) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{n} x_i^2}\right) - 0.5}{\left(1 + 0.001 * \left[\sum_{i=1}^{n} x_i^2\right]\right)^2}$
F22	Himmelblau	$[-5, 5]^D$	MS	$f_{22}\left(\vec{x}\right) = \frac{1}{n}\sum_{i=1}^{n}\left(x_i^4 - 16x_i^2 + 5x_i\right)$
F23	Michalewicz	$[0, \pi]^D$	MS	$f_{23}\left(\vec{x}\right) = -\sum_{i=1}^{n}\sin(x_i)\sin^{20}\left(\frac{ix_i^2}{\pi}\right)$

Table 1.
Benchmark functions.

4. Experimental results

This section presents the experimental study for testing the performance of proposed algorithm in comparison to a number of state-of-the-art algorithms. To fairly compare the performance of proposed approach, each algorithm was executed with 30 different runs; each seeded randomly. The algorithmic parameters of proposed approach and other variants, AP, FL, N are set to $0.1, 2, 30$, respectively. The maximum function evaluation is set to $1.5E + 5$. Firstly, proposed approach is compared with the original CSA in **Table 2**. After that, CSA-MSS is compared with all strategies in **Tables 3–6**. The parameters of strategy selection scheme, i.e. Adaptive Pursuit, are set to $\alpha = 0.1$, $beta = 0.1$, the reward value is calculated with extreme approach, i.e. maximum, over the last $W = 5$ iterations. All parametric settings are delivered using parameter tuning experiments.

Table 2 shows the results of CSA and CSA-MSS algorithms on 30- and 60-dimensional benchmark functions. For the 30D problems, CSA looks slightly competing with the CSA-MSS. The column with *Sign* headings represents Wilcoxon signed sum rank test result. If the sign is $+$, it means that the result is statistically sound, otherwise not sound, where the results of four out of 23 functions do not look statistically sound. In eight out of 19 functions, CSA produces better solutions while, in 11 out 19 functions, CSA-MSS performs better. When we look at the characteristics of the functions best obtained with CSA, it is clearly shown that most of them are multi-modal, because CSA has good performance on these types of problems. On the other side, CSA-MSS has improved the success over the CSA on uni-modal functions. For the 60D functions, CSA-MSS outperforms CSA in both mean and standard deviation statistics of the best solutions. And the success of CSA-MSS is statistically sound for all benchmark functions.

The comparative experimental results with 30 D benchmark functions for different search strategies are presented in **Tables 3** and **4**. CSA-MSS has produced the minimum mean values for F1, F2, F3, F4, F5, F8, F9, F10, F14 functions, while CSA achieves slightly better results for F7, F13, F15, F16, F20, F21 functions. For F6, F7, F12, F17, F19 functions, GCSA produced better solutions than the others. RMCSA found the minimum mean values for F11, F18, F22, F23 functions, while CBCSA and RCSA have lower performance than the others presumably due to their poorer exploitation capabilities. The tables for both dimensions are combined to conduct statistical confidence test of the success, which demonstrated that the success of CSA-MSS over the others is found statistically sound with an exemption that the success of RMCSA for the functions F11 and F18 is not statistically different from CSA-MSS. It is noted that the mean rank values by CSA-MSS remain slightly better than the other methods.

The comparison of the experimental results for 60 D benchmark functions with different search strategies are provided in **Tables 5** and **6**. CSA-MSS has the minimum mean values except for F6, F7, F10, F11, F12, F13, F15, F19, F20 functions. Nevertheless, for all functions considered in the test, CSA-MSS produced better solutions than CSA. It is also observed that GCSA produced better solutions than CSA-MSS for F6, F7, F10, F12, F13, F19 functions, and RMCSA produced better solutions than CSA-MSS for F11, F15, F20 functions. CSA turns to poorer performance when the problem dimension increases. The statistical soundness of the success of CSA-MSS is approved except for F21, F22 functions. For the comparison of mean rank values, CSA-MSS outperforms the other methods.

Figure 1 depicts the convergence comparison of the methods over the iterations. As seen, CSA-MSS has higher convergence speed from the others for F1,F2,F3 and F4 functions. For the 60 D problems, **Figure 2** shows the convergence history for the all methods.

	30D					**60D**				
F. No	**CSA**		**CSA-MSS**			**CSA**		**CSA-MSS**		
	Mean	**Std**	**Mean**	**Std**	**Sign**	**Mean**	**Std**	**Mean**	**Std**	**Sign**
F1	4.77E-05	2.81E-05	**1.79E-74**	**6.79E-74**	+	7.87E+04	1.99E+04	**7.63E-17**	**3.10E-16**	+
F2	3.21E+00	5.95E+00	**3.18E−66**	**1.70E-65**	+	2.71E+09	4.69E+08	**1.27E-12**	**5.75E-12**	+
F3	6.60E-06	4.00E-06	**1.78E-69**	**9.59E-69**	+	2.50E+04	5.06E+03	**9.11E-18**	**4.79E-17**	+
F4	6.71E-14	1.21E-13	**1.62E-78**	**8.73E-78**	+	5.81E+40	1.37E+41	**3.72E−31**	**2.00E-30**	+
F5	1.32E-03	2.78E-04	**6.17E-34**	**2.98E-33**	+	3.17E+18	8.63E+18	**1.74E-08**	**9.13E-08**	+
F6	**5.72E-02**	1.95E-02	4.37E-01	1.89E-01	+	8.35E+00	2.73E-01	**2.94E+00**	**4.17E-01**	+
F7	**0.00E+00**	0.00E+00	3.30E+00	2.15E+00	+	7.75E+04	2.16E+04	**3.15E+01**	**1.94E+01**	+
F8	5.18E-14	8.21E-14	**7.10E−62**	**3.82E-61**	+	2.39E+02	7.05E+01	**2.70E-31**	**9.43E-31**	+
F9	1.91E-02	5.43E-03	**9.85E-03**	**4.71E-03**	+	2.16E+02	7.95E+01	**5.40E-02**	**1.41E-02**	+
F10	2.74E+01	9.95E+00	**8.48E+00**	**6.43E+00**	+	2.97E+06	8.81E+05	**9.13E+01**	**3.25E+01**	+
F11	**3.73E+01**	**1.37E+01**	5.01E+01	1.76E+01	+	7.86E+02	3.09E+01	**1.16E+02**	**1.62E+01**	+
F12	5.47E+01	2.29E+01	**5.37E+01**	**1.67E+01**	−	7.49E+02	2.83E+01	**1.18E+02**	**2.70E+01**	+
F13	**6.30E-03**	**7.08E-03**	3.93E-02	4.93E-02	+	6.57E+02	2.23E+02	**8.54E-02**	**1.16E-01**	+
F14	8.37E+03	3.37E+02	**4.38E+03**	**7.61E+02**	+	1.92E+04	3.24E+02	**1.00E+04**	**1.07E+03**	+
F15	**1.40E-03**	**3.93E-04**	3.24E+00	1.28E+00	+	1.98E+01	9.61E-01	**6.89E+00**	**1.48E+00**	+
F16	**3.78E-07**	**2.89E-07**	2.08E-01	4.50E-01	−	4.63E+08	1.41E+08	**3.76E-01**	**6.87E-01**	+
F17	**4.60E-06**	**3.78E-06**	4.84E-03	2.61E-02	+	8.62E+08	3.08E+08	**3.33E-02**	**1.62E-01**	+
F18	1.19E-03	1.03E-03	**2.17E-09**	**1.17E-08**	+	1.08E+02	8.18E+00	**3.14E-05**	**1.69E-04**	+
F19	**3.19E-06**	**2.84E-06**	4.47E-01	1.26E+00	−	1.16E+03	3.39E+02	**7.76E-01**	**1.84E+00**	+
F20	**7.41E-02**	**1.82E-02**	7.75E+00	2.28E+00	+	8.90E+01	5.61E+00	**2.81E+01**	**3.89E+00**	+
F21	**5.73E-02**	**2.02E-02**	8.01E-02	1.45E-02	+	3.97E-01	8.65E-03	**1.66E-01**	**2.17E-02**	+
F22	**-6.89E+01**	**2.17E+00**	-6.88E+01	2.88E+00	−	-3.65E+01	1.79E+00	**-6.72E+01**	**1.55E+00**	+
F23	−1.56E+01	3.43E+00	**−2.58E+01**	**1.20E+00**	+	−1.47E+01	8.09E-01	**−5.04E+01**	**2.53E+00**	+

Table 2.
The comparison of CSA and CSA-MSS on benchmark functions.

F No	CSA		GCSA		BCSA		CBCSA		RCSA		RMCSA		CSA-MSS	
	Mean	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**
F1	4.77E-05	2.81E-05	1.75E-19	1.45E-19	3.63E+04	1.22E+04	5.90E+03	5.21E+02	3.58E+04	5.30E+03	1.55E-39	6.04E-39	**1.79E-74**	**6.79E-74**
F2	3.21E+00	5.95E+00	4.48E-09	1.25E-08	6.03E+08	1.45E+08	7.49E+07	1.23E+07	5.83E+08	9.55E+07	1.36E-34	5.01E-34	**3.18E−66**	**1.70E−65**
F3	6.60E-06	4.00E-06	4.06E-20	3.41E-20	5.02E+03	1.18E+03	7.96E+02	1.06E+02	5.23E+03	7.18E+02	5.20E-39	2.21E-38	**1.78E-69**	**9.59E-69**
F4	6.71E-14	1.21E-13	4.82E-17	5.72E-17	1.32E+17	2.45E+17	1.80E+08	1.78E+08	5.81E+16	1.13E+17	8.72E+01	4.25E+02	**1.62E-78**	**8.73E-78**
F5	1.32E-03	2.78E-04	4.68E-11	2.34E-11	1.51E+06	2.29E+06	3.75E+01	3.23E+00	9.98E+05	1.83E+06	8.04E-27	3.13E-26	**6.17E-34**	**2.98E-33**
F6	5.72E-02	1.95E-02	**2.10E-02**	**2.29E-02**	7.11E+00	2.23E-01	3.23E+00	1.92E-01	7.06E+00	1.88E-01	2.30E+00	4.71E-01	4.37E-01	1.89E-01
F7	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.03E+04	4.31E+03	5.79E+03	7.46E+02	3.58E+04	4.35E+03	1.34E+01	3.25E+01	3.30E+00	2.15E+00
F8	5.18E-14	8.21E-14	7.12E-39	1.65E-38	4.39E+01	1.09E+01	1.24E+00	2.28E-01	3.48E+01	1.02E+01	4.80E-38	2.55E-37	**7.10E-62**	**3.82E-61**
F9	1.91E-02	5.43E-03	1.63E-02	5.13E-03	4.69E+01	1.18E+01	1.56E+00	2.75E-01	3.55E+01	7.44E+00	1.85E-02	1.88E-02	**9.85E-03**	**4.71E-03**
F10	2.74E+01	9.95E+00	2.85E+01	2.17E+01	1.05E+06	4.52E+05	3.30E+04	7.35E+03	9.65E+05	2.20E+05	2.65E+01	2.01E+01	**8.48E+00**	**6.43E+00**
F11	3.73E+01	1.37E+01	4.75E+01	1.08E+01	3.41E+02	1.14E+01	2.22E+02	1.19E+01	3.29E+02	1.64E+01	**3.58E+01**	**7.61E+00**	5.01E+01	1.76E+01
F12	5.47E+01	2.29E+01	**2.97E+01**	**8.55E+00**	3.04E+02	1.50E+01	1.89E+02	9.60E+00	2.92E+02	2.00E+01	5.19E+01	1.41E+01	5.37E+01	1.67E+01
F13	**6.30E-03**	**7.08E-03**	1.73E-02	3.45E-02	3.62E+02	3.72E+01	5.42E+01	7.96E+00	3.29E+02	3.57E+01	1.33E-02	1.70E-02	3.93E-02	4.93E-02
F14	8.37E+03	3.37E+02	5.51E+03	6.15E+02	8.40E+03	2.89E+02	8.31E+03	2.03E+02	8.39E+03	2.91E+02	4.51E+03	1.52E+03	**4.38E+03**	**7.61E+02**
F15	**1.40E-03**	**3.93E-04**	3.87E-02	2.08E-01	1.97E+01	5.59E-01	1.36E+01	4.54E-01	1.95E+01	2.02E-01	9.61E-01	7.36E-01	3.24E+00	1.28E+00
F16	**3.78E-07**	**2.89E-07**	2.81E-01	1.33E+00	1.82E+08	3.58E+07	1.81E+05	8.24E+04	1.26E+08	3.23E+07	3.90E-01	8.09E-01	2.08E-01	4.50E-01
F17	4.60E-06	3.78E-06	**3.74E-19**	**5.71E-19**	3.60E+08	1.13E+08	2.90E+06	1.02E+06	2.88E+08	8.23E+07	2.90E-01	9.00E-01	4.84E-03	2.61E-02
F18	1.19E-03	1.03E-03	2.49E-04	7.46E-04	4.53E+01	2.61E+00	2.32E+01	1.19E+00	4.33E+01	2.31E+00	**1.67E-15**	**1.36E-15**	2.17E-09	1.17E-08
F19	3.19E-06	2.84E-06	**2.65E-13**	**3.97E-13**	5.58E+02	7.14E+01	8.59E+01	1.02E+01	5.08E+02	6.01E+01	9.84E-02	1.63E-01	4.47E-01	1.26E+00
F20	**7.41E-02**	**1.82E-02**	1.14E+01	3.19E+00	4.16E+01	4.07E+00	2.72E+01	7.16E-01	4.12E+01	1.63E+00	8.72E-01	7.99E-01	7.75E+00	2.28E+00
F21	**5.73E-02**	**2.02E-02**	1.14E-01	2.52E-02	2.76E-01	3.35E-03	1.10E-01	1.99E-02	2.63E-01	1.88E-02	6.54E-02	2.82E-02	8.01E-02	1.45E-02
F22	-6.89E+01	2.17E+00	-6.76E+01	2.75E+00	−4.54E+01	4.79E+00	−5.29E+01	1.49E+00	−4.46E+01	1.62E+00	**−7.12E+01**	**2.17E+00**	−6.88E+01	2.88E+00
F23	−1.56E+01	3.43E+00	−2.56E+01	1.24E+00	−9.43E+00	6.27E-01	−1.16E+01	4.97E-01	−9.32E+00	3.83E-01	**−2.61E+01**	**1.08E+00**	−2.58E+01	1.20E+00

Table 3.
The comparison of the search strategies on 30D benchmark functions.

F No	CSA		GCSA		BCSA		CBCSA		RCSA		RMCSA		CSA-MSS
	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank
F1	4	+	3	+	7	–	5	–	6	–	2	+	1
F2	4	–	3	+	7	–	5	–	6	–	2	+	1
F3	4	+	3	+	6	–	5	–	7	–	2	+	1
F4	3	+	2	+	7	–	5	–	6	–	4	–	1
F5	4	+	3	+	7	–	5	–	6	–	2	+	1
F6	2	+	1	+	7	–	5	–	6	–	4	–	3
F7	1	+	1	+	7	–	5	–	6	–	4	–	3
F8	4	+	2	+	7	–	5	–	6	–	3	+	1
F9	4	+	2	+	7	–	5	–	6	–	3	+	1
F10	3	–	4	–	7	–	5	–	6	–	2	–	1
F11	2	–	3	–	7	–	5	–	6	–	1	–	4
F12	4	–	1	–	7	–	5	–	6	–	2	–	3
F13	1	+	3	+	7	–	5	–	6	–	2	+	4
F14	5	–	3	–	7	–	4	–	6	–	2	–	1
F15	1	+	2	–	7	–	5	–	6	–	3	–	4
F16	1	+	3	–	7	–	5	–	6	–	4	–	2
F17	2	+	1	+	7	–	5	–	6	–	4	–	3
F18	4	+	3	+	7	–	5	–	6	–	1	+	2
F19	2	+	1	+	7	–	5	–	6	–	3	–	4
F20	1	+	4	–	7	–	5	–	6	–	2	–	3
F21	1	+	5	+	7	+	4	+	6	+	2	+	3
F22	2	–	4	–	6	–	5	–	7	–	1	–	3
F23	4	–	3	–	6	–	5	–	7	–	1	–	2
Mean:	2.73913		2.608696		6.869565		4.913043		6.130435		2.434783		2.26087

Table 4.
The statistical details of comparison of the search strategies on 30D benchmark functions.

F No	CSA		GCSA		BCSA		CBCSA		RCSA		RMCSA		CSA-MSS	
	Mean	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**
F1	7.87E+04	1.99E+04	8.51E-13	4.83E-13	1.13E+05	4.89E+03	2.90E+04	2.32E+03	1.14E+05	5.66E+03	5.27E-09	2.05E-08	**7.63E-17**	**3.10E-16**
F2	2.71E+09	4.69E+08	2.66E+00	3.27E+00	2.88E+09	4.26E+08	6.08E+08	1.02E+08	2.83E+09	3.31E+08	2.83E+08	7.80E+08	**1.27E-12**	**5.75E-12**
F3	2.50E+04	5.06E+03	6.64E-13	3.17E-13	3.20E+04	1.42E+03	8.04E+03	5.22E+02	3.19E+04	2.11E+03	1.08E-08	4.21E-08	**9.11E-18**	**4.79E-17**
F4	5.81E+40	1.37E+41	3.26E-14	2.84E-14	2.11E+41	9.55E+41	6.83E+27	1.42E+28	7.94E+40	1.43E+41	2.38E+40	6.95E+40	**3.72E−31**	**2.00E-30**
F5	3.17E+18	8.63E+18	2.37E-07	1.04E-07	6.61E+19	1.69E+20	4.04E+02	4.16E+02	5.83E+19	1.16E+20	1.62E+17	7.84E+17	**1.74E-08**	**9.13E-08**
F6	8.35E+00	2.73E-01	**1.49E+00**	**2.84E-01**	8.41E+00	2.04E-01	5.37E+00	1.71E-01	8.42E+00	2.01E-01	4.96E+00	1.73E+00	2.94E+00	4.17E-01
F7	7.75E+04	2.16E+04	**2.00E-01**	**5.42E-01**	1.15E+05	5.48E+03	3.03E+04	1.29E+03	1.15E+05	4.54E+03	1.88E+02	1.64E+02	3.15E+01	1.94E+01
F8	2.39E+02	7.05E+01	1.24E-26	9.93E-27	3.46E+02	3.04E+01	3.03E+01	3.88E+00	3.41E+02	2.53E+01	8.70E-12	4.44E-11	**2.70E-31**	**9.43E-31**
F9	2.16E+02	7.95E+01	7.06E-02	1.91E-02	3.53E+02	3.04E+01	2.86E+01	4.47E+00	3.45E+02	3.60E+01	1.75E-01	5.41E-02	**5.40E-02**	**1.41E-02**
F10	2.97E+06	8.81E+05	**8.99E+01**	**4.53E+01**	4.76E+06	5.80E+05	4.18E+05	5.10E+04	4.74E+06	4.95E+05	1.52E+02	4.80E+01	9.13E+01	3.25E+01
F11	7.86E+02	3.09E+01	1.14E+02	2.44E+01	8.17E+02	1.68E+01	5.72E+02	1.96E+01	8.07E+02	2.57E+01	**1.05E+02**	**6.56E+01**	1.16E+02	1.62E+01
F12	7.49E+02	2.83E+01	**1.04E+02**	**2.23E+01**	7.54E+02	2.62E+01	5.20E+02	2.25E+01	7.55E+02	3.01E+01	3.09E+02	2.95E+02	1.18E+02	2.70E+01
F13	6.57E+02	2.23E+02	**6.16E-03**	**7.24E-03**	1.03E+03	5.50E+01	2.69E+02	1.64E+01	1.04E+03	4.44E+01	1.72E-02	3.56E-02	8.54E-02	1.16E-01
F14	1.92E+04	3.24E+02	1.43E+04	1.87E+03	1.91E+04	5.35E+02	1.92E+04	4.43E+02	1.92E+04	4.70E+02	1.93E+04	4.05E+02	**1.00E+04**	**1.07E+03**
F15	1.98E+01	9.61E-01	7.41E+00	2.29E+00	2.04E+01	6.71E-02	1.69E+01	2.91E-01	2.04E+01	9.49E-02	**4.52E+00**	**1.44E+00**	6.89E+00	1.48E+00
F16	4.63E+08	1.41E+08	6.56E-01	7.70E-01	7.97E+08	8.99E+07	2.08E+07	6.58E+06	7.88E+08	1.03E+08	2.00E+04	1.03E+05	**3.76E-01**	**6.87E-01**
F17	8.62E+08	3.08E+08	7.58E+00	1.25E+01	1.59E+09	1.37E+08	8.28E+07	1.47E+07	1.58E+09	1.87E+08	5.30E+03	2.59E+04	**3.33E-02**	**1.62E-01**
F18	1.08E+02	8.18E+00	1.09E-02	1.20E-02	1.11E+02	4.63E+00	6.81E+01	2.56E+00	1.13E+02	4.00E+00	4.14E+01	5.45E+01	**3.14E-05**	**1.69E-04**
F19	1.16E+03	3.39E+02	**3.66E-03**	**1.97E-02**	1.67E+03	8.70E+01	4.58E+02	2.73E+01	1.68E+03	7.68E+01	4.64E-01	1.22E+00	7.76E-01	1.84E+00
F20	8.90E+01	5.61E+00	3.77E+01	4.39E+00	9.61E+01	1.37E+00	6.81E+01	1.64E+00	9.56E+01	1.64E+00	**1.02E+01**	**3.15E+00**	2.81E+01	3.89E+00
F21	3.97E-01	8.65E-03	2.25E-01	2.60E-02	4.07E-01	7.33E-03	2.57E-01	1.87E-02	4.04E-01	7.64E-03	1.71E-01	2.17E-02	**1.66E-01**	**2.17E-02**
F22	-3.65E+01	1.79E+00	−6.65E+01	1.54E+00	−3.57E+01	1.35E+00	−4.34E+01	1.07E+00	−3.60E+01	1.66E+00	−6.49E+01	1.08E+01	**−6.72E+01**	**1.55E+00**
F23	−1.47E+01	8.09E-01	−4.73E+01	1.75E+00	−1.49E+01	7.20E-01	−1.80E+01	7.08E-01	−1.46E+01	5.62E-01	−1.48E+01	7.80E-01	**−5.04E+01**	**2.53E+00**

Table 5.
The comparison of the search stragies on 60D benchmark functions.

F No	CSA		GCSA		BCSA		CBCSA		RCSA		RMCSA		CSA-MSS
	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank	Sign	Rank
F1	5	+	2	+	6	+	4	+	7	+	3	+	1
F2	5	+	2	+	7	+	4	+	6	+	3	+	1
F3	5	+	2	+	7	+	4	+	6	+	3	+	1
F4	5	+	2	+	7	+	3	+	6	+	4	+	1
F5	5	+	2	+	7	+	3	+	6	+	4	+	1
F6	5	+	1	+	6	+	4	+	7	+	3	+	2
F7	5	+	1	+	7	+	4	+	6	+	3	+	2
F8	5	+	2	+	7	+	4	+	6	+	3	+	1
F9	5	+	2	+	7	+	4	+	6	+	3	+	1
F10	5	+	1	+	7	+	4	+	6	+	3	+	2
F11	5	+	2	+	7	+	4	+	6	+	1	+	3
F12	5	+	1	+	6	+	4	+	7	+	3	+	2
F13	5	+	1	+	6	+	4	+	7	+	2	+	3
F14	6	+	2	+	3	+	4	+	5	+	7	+	1
F15	5	+	3	+	7	+	4	+	6	+	1	+	2
F16	5	+	2	+	7	+	4	+	6	+	3	+	1
F17	5	+	2	+	7	+	4	+	6	+	3	+	1
F18	5	+	2	+	6	+	4	+	7	+	3	+	1
F19	5	+	1	+	6	+	4	+	7	+	2	+	3
F20	5	+	3	+	7	+	4	+	6	+	1	+	2
F21	5	+	3	+	7	+	4	+	6	+	2	+	1
F22	5	+	2	+	7	+	4	+	6	+	3	+	1
F23	6	+	2	+	4	+	3	+	7	+	5	+	1
Mean:	5.086957		1.869565		6.434783		3.869565		6.26087		2.956522		1.521739

Table 6.
The statistical details of comparison of the search stragies on 60D benchmark functions.

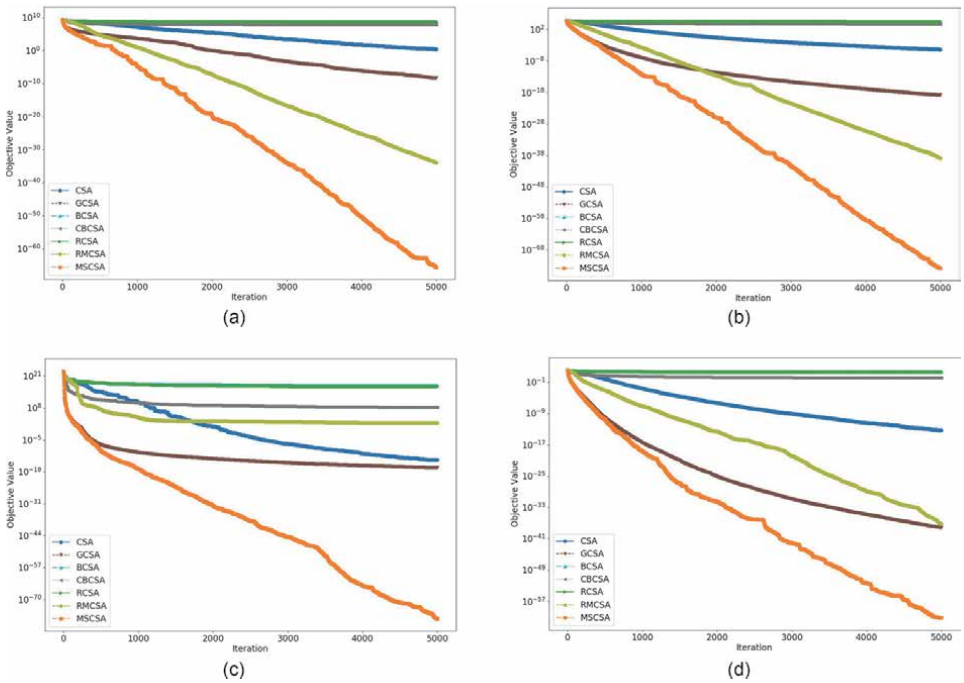


Figure 1.
The converge graphs of compared methods on 30D benchmark functions. (a) F1 Function. (b) F2 Function. (c) F4 Function. (d) F8 Function.

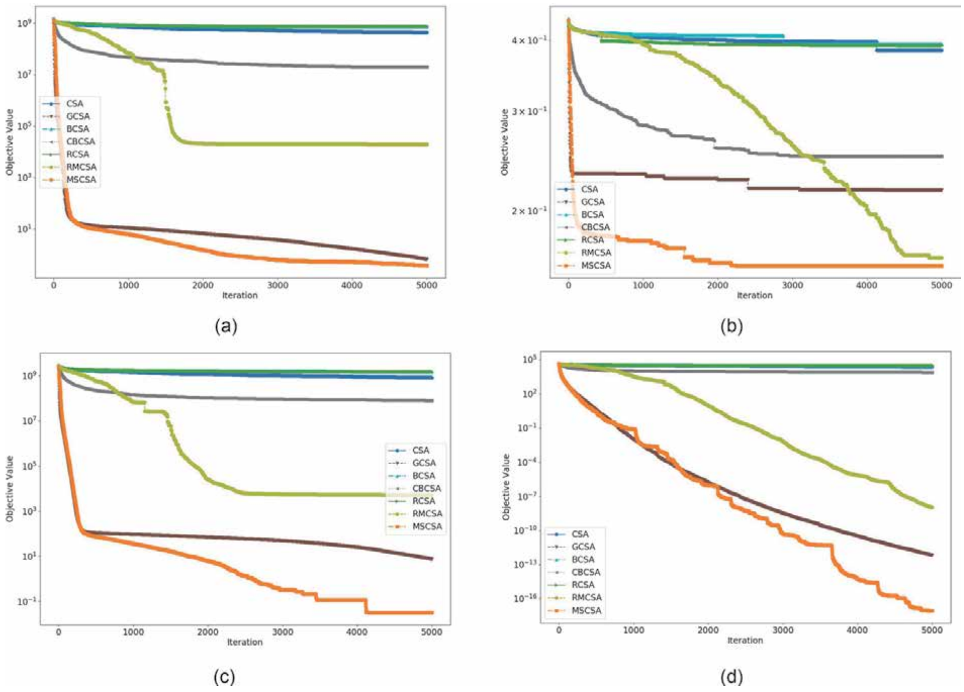


Figure 2.
The converge graphs of compared methods on 60D benchmark functions. (a) F16 Function. (b) F21 Function. (c) F17 Function. (d) F3 Function.

5. Conclusions

In this article, a multi-strategy search with crow search algorithm is proposed for solving global optimisation problems. In the proposed approach, adaptive pursuit is used to select search strategies according to the search dynamics. Search strategies are obtained from other state-of-the-art metaheuristics and implemented into crow search algorithm. Twenty-three benchmark functions with different characteristics have been used to compare strategies' performance. The proposed approach is demonstrated that it helps improve the performance of original crow search algorithm especially on the higher-dimensional problems. Future works of this study would be dynamic and adaptive parametric settings, especially for awareness probability and flight length distance. It may also include embedding machine learning approaches to develop high-performance adaptive strategy selection schemes into the algorithm to achieve transfer learning.

Acknowledgements

This study has been conducted under the research support scheme of 2219, namely "Post-doctorate research programme", TUBITAK of Republic of Turkey.

Conflict of interest

The authors declare no conflict of interest.

Notes/thanks

The first author would like to show his thanks to TUBITAK for supporting his visit in UWE Bristol, UK, in the period between October, 2021 and October 2022.

Author details

Rafet Durgut[1,2*†] and Mehmet Emin Aydin[2†]

1 Computer Engineering, Bandirma Onyedi Eylul University, Balikesir, Turkey

2 UWE, Computer Science and Creative Technologies, Bristol, UK

*Address all correspondence to: rdurgut@bandirma.edu.tr;
rafet.durgut@uwe.ac.uk

† These authors contributed equally.

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. cc BY

References

[1] Askarzadeh A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. Computers & Structures. 2016;**169**:1-12

[2] Said GAE-NA, Mahmoud AM, El-Horbaty E-SM. A comparative study of meta-heuristic algorithms for solving quadratic assignment problem. arXiv preprint arXiv: 1407.4863. 2014

[3] Helmi AM et al. Efficient and sustainable reconfiguration of distribution networks via metaheuristic optimization. In: IEEE Transactions on Automation Science and Engineering. IEEE; 2021

[4] Kalita K, Ghadai RK, Chakraborty S. A comparative study on the metaheuristic-based optimization of skew composite laminates. Engineering with Computers. 2021:1-18. (In press)

[5] Baatar N, Zhang D, Koh C-S. An improved differential evolution algorithm adopting \?-best mutation strategy for global optimization of electromagnetic devices. IEEE Transactions on Magnetics. 2013;**49**(5): 2097-2100

[6] Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya, Japan: IEEE; 1995. pp. 39-43

[7] Wang F, Zhang H, Zhou A. A particle swarm optimization algorithm for mixed-variable optimization problems. Swarm and Evolutionary Computation. 2021;**60**:100808

[8] Durgut R, Aydin ME. Adaptive binary artificial bee colony algorithm. Applied Soft Computing. 2021;**101**: 107054

[9] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing. 2008;**8**(1):687-697

[10] Garcıa J, Maureira C. A KNN quantum cuckoo search algorithm applied to the multidimensional knapsack problem. Applied Soft Computing. 2021;**102**:107077

[11] Yang X-S, Deb S. Cuckoo search via Lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). IEEE; 2009. pp. 210-214

[12] Naik A, Satapathy SC. A comparative study of social group optimization with a few recent optimization algorithms. Complex & Intelligent Systems. 2021;**7**(1):249-295

[13] He S, Wu QH, Saunders JR. Group search optimizer: An optimization algorithm inspired by animal searching behavior. IEEE Transactions on Evolutionary Computation. 2009;**13**(5): 973-990

[14] Yang X-S. Firefly algorithm, levy flights and global optimization. In: Research and Development in Intelligent Systems XXVI. Springer; 2010. pp. 209-218

[15] Yang X-S, Slowik A. Firefly algorithm. In: Swarm Intelligence Algorithms. CRC Press; 2020. pp. 163-174

[16] Morales-Castañeda B et al. A better balance in metaheuristic algorithms: Does it exist? Swarm and Evolutionary Computation. 2020;**54**:100671

[17] Hughes M, Goerigk M, Dokka T. Particle swarm metaheuristics for robust optimisation with implementation uncertainty. Computers

& Operations Research. 2020;**122**: 104998

[18] Majhi SK, Sahoo M, Pradhan R. A space transformational crow search algorithm for optimization problems. Evolutionary Intelligence. 2020;**13**(3): 345-364

[19] Ozsari S et al. Adaptation of metaheuristic algorithms to improve training performance of an ESZSL model. Turkish Journal of Electrical Engineering & Computer Sciences. 2021;**29**:3

[20] Xiao S et al. Artificial bee colony algorithm based on adaptive neighborhood search and Gaussian perturbation. Applied Soft Computing. 2021;**100**:106955

[21] Hu P et al. Multi-strategy serial cuckoo search algorithm for global optimization. Knowledge-Based Systems. 2021;**214**:106729

[22] Zamani H, Nadimi-Shahraki MH, Gandomi AH. CCSA: Conscious neighborhood-based crow search algorithm for solving global optimization problems. Applied Soft Computing. 2019;**85**:105583

[23] Sonuc E. Binary crow search algorithm for the uncapacitated facility location problem. Neural Computing and Applications. 2021;**33**(21):1-17

[24] Majhi SK, Sahoo M, Pradhan R. Oppositional crow search algorithm with mutation operator for global optimization and application in designing FOPID controller. Evolving Systems. 2019;**12**(2):1-26

[25] Rizk-Allah RM, Hassanien AE, Bhattacharyya S. Chaotic crow search algorithm for fractional optimization problems. Applied Soft Computing. 2018;**71**:1161-1175

[26] dos Santos Coelho L et al. Electromagnetic optimization based on Gaussian crow search approach. In: 2018 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM). IEEE; 2018. pp. 1107-1112

[27] Necira A et al. Dynamic crow search algorithm based on adaptive parameters for large-scale global optimization. Evolutionary Intelligence. 2021:1-17. (In press)

[28] Hussien AG et al. Crow search algorithm: Theory, recent advances, and applications. IEEE Access. 2020;**8**: 173548-173565

[29] Meraihi Y et al. A comprehensive survey of crow search algorithm and its applications. Artificial Intelligence Review. 2020;**54**(4):1-48

[30] Prior H, Schwarz A, Güntürkün O. Mirror-induced behavior in the magpie (Pica pica): Evidence of self-recognition. PLoS Biology. 2008;**6**(8): e202

[31] Black H. Social skills to crow about. Scientific American Mind. 2013;**24**(4): 12-12

[32] Abdelaziz AY, Fathy A. A novel approach based on crow search algorithm for optimal selection of conductor size in radial distribution networks. Engineering Science and Technology, an International Journal. 2017;**20**(2):391-402

[33] Kiran MS et al. Artificial bee colony algorithm with variable search strategy for continuous optimization. Information Sciences. 2015;**300**:140-157

[34] Peng H, Deng C, Zhijian W. Best neighbor-guided artificial bee colony algorithm for continuous optimization problems. Soft Computing. 2019;**23**(18): 8723-8740

[35] Liang J, Xue Y. An adaptive GP-based memetic algorithm for symbolic

regression. Applied Intelligence. 2020; **50**(11):3961-3975

[36] Aydin ME. Coordinating metaheuristic agents with swarm intelligence. Journal of Intelligent Manufacturing. 2012;**23**(4):991-999

[37] Abualigah L et al. A parallel hybrid krill herd algorithm for feature selection. International Journal of Machine Learning and Cybernetics. 2021;**12**(3):783-806

[38] Durgut R, Yavuz İ, Aydin M. Küme Birles ›imli Sırt Çantası Probleminin Adaptif Yapay Arı Kolonisi Algoritması ile Çözümü. Journal of Intelligent Systems: Theory and Applications. 2021: 43-54. DOI: 10.38016/jista.854584

[39] Surjanovic S, Bingham D. Virtual Library of Simulation Experiments: Test Functions and Datasets. Burnaby, BC, Canada: Simon Fraser University; 2013. p. 2015. [Accessed: May 13]

[40] Zhu G, Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization. Applied Mathematics and Computation. 2010;**217**(7):3166-3173

[41] Trelea IC. The particle swarm optimization algorithm: Convergence analysis and parameter selection. Information Processing Letters. 2003; **85**(6):317-325

[42] Babaoglu I. Artificial bee colony algorithm with distribution-based update rule. Applied Soft Computing. 2015;**34**:851-861

Chapter 2

Hybrid Genetic Algorithms

Leila Gharsalli

Abstract

Hybrid optimization methods have known significant interest in recent years and are being growingly used to solve complex problems in science and engineering. For instance, the famous evolutionary Genetic Algorithm can integrate other techniques within its framework to produce a hybrid global algorithm that takes advantages of that combination and overcomes the disadvantages. Several forms of integration between Genetic Algorithms and other search and optimization techniques exist. This chapter aims to review that and present the design of a hybrid Genetic Algorithm incorporating another local optimization technique while recalling the main local search methods and emphasizing the different approaches for employing their information. A test case from the aerospace field is presented where a hybrid genetic algorithm is proposed for the mechanical sizing of a composite structure located in the upper part of a launcher.

Keywords: genetic algorithm, evolutionary algorithm, hybridization, local search, mechanical sizing, aerospace field

1. Introduction

Solving an optimization problem consists in exploring a search space in order to maximize (or minimize) a given objective function. The complexities (in size or structure) of the search space and the function to be optimized lead to the use of radically different resolution methods. However, the so-called global optimization methods are generally the best suited ones considering their main advantage: that of identifying the promised regions and converging towards a global optimum.

The zero-order optimization Genetic Algorithm (GA), initially laid down by Holland [1], remain the most recognized and practiced form of Evolutionary Algorithms (EA) which are global stochastic optimization techniques that mimic Darwin's principles of natural selection and genetic dynamics. In fact, known advantages of the GA include its ability to combine both exploration and exploitation in an optimal way [1] in addition to its ability to be used in the case of discontinuous objective functions, within disjoined and/or non-convex design spaces, and together with discrete, continuous, or even mixed design variables. Besides, as a population-based method, GA, if initially well-tuned, minimizes the risk to converge to a local optimum thanks to the simultaneous processing of the whole candidate solutions instead of, for instance, gradient-based methods, giving thus the designer a multitude of options.

Nevertheless, like any optimization technique, GA has also its drawbacks starting with the numerous parameters sensitivity analysis to do to maximize efficiency and have a good initialization of the algorithm. In addition to the most notable drawback which is generally the great number of required iterations, and they slow convergence, especially in the neighborhood of the global optimum. Indeed, although GAs can rapidly locate the region in which the global optimum exists, they take a relatively long time to locate the exact local optimum in the region of convergence. Each generation of the algorithm consists of many objective function evaluations resulting in a lot of computational time in addition to being very expensive.

To overcome this main limitation and lessen function evaluation time, different alternatives have been proposed. But particularly the trend of hybridization has been observed in many works carried out on metaheuristics over the past 20 years, where it consists of incorporating a faster Local Search (LS) (or individual learning) optimization algorithm into a GA in order to improve the performance of the global resulting method, often known as Memetic Algorithm (MA). This latter was first introduced by Moscato [2] and is representing one of the recent growing areas of research in evolutionary computation [3] by offering the important challenge of the tradeoff between global searching and local searching in terms of time and computational effort. In fact, this integration aims to keep advantages of both optimization methods while offsetting their both disadvantages [4] As LS algorithms are actually capable to find the local optimum with high accuracy and fast convergence but suffer from the problem of foothills. Hence through their integration with the population-based method, the time needed to reach the global optimum can be further reduced since the local knowledge is used to accelerate finding the most promising search region in addition to locating the global optimum starting within its attraction basin.

These hybrid algorithms have also been used under the name of hybrid evolutionary algorithms, Baldwinian evolutionary algorithms, Lamarckian evolutionary algorithms, cultural algorithms, or genetic LS. They also have been successfully applied to hundreds of real-world problems in a wide variety of domains ranging from aircraft [5], aerospace [6], pattern recognition [7], control systems [8], vehicle routing [9], pharmaceutical industry [10] etc.

An example of a hybrid GA applied to the mechanical structures' optimization is presented in this chapter. This test case reflects the relevance of this hybridization in the mathematical resolution of optimization problems related to the aerospace field.

The organization of the chapter is as follows: the second section is dedicated to the description of the different hybridization design including the combination between GAs and LS methods. General mechanisms of GAs are remained in the third section. Hybrid GAs are discussed in the fourth section then the test case with the problematic presentation and the obtained results is given in the fifth section. Finally, conclusions are given in the last section.

2. Hybridization design

As mentioned before, in an optimization problem solving approach, it can be extremely beneficial to combine two solving techniques, knowing that it is possible to hybridize all exact and metaheuristic techniques. Hence, the hybridization can take place in one or more components of a research method. It can also consist in assembling several hybridization methods to form a single hybrid method. Two strategies can be used depending on the optimization technique which drives the hybrid method:

Strategy 1: we can approach the global optimum by a stochastic technique and then refine the result by successively applying a local one. In this case, the result will be ameliorated but unfortunately time consuming. An optimal way to use

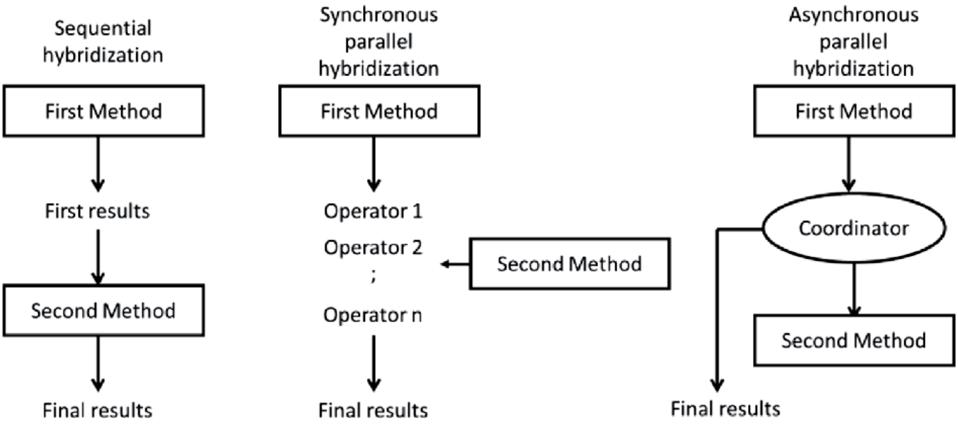


Figure 1.
Hybridization classification.

hybridization in this case may be by applying a global optimization method and find the right time to switch to a local one.

Strategy 2: we can try to find a local minimum and keep it as a winning solution if it is the best of all the local (global) minimums. For instance, in the method of multiple initializations or (multi Start), a local optimization technique is used several times at different stress points; the solution of the optimization problem would be then the best obtained result.

According to the author in [11], a taxonomy for hybrid meta-heuristics can be proposed by classifying them into three categories depending on their architecture (see **Figure 1**):

- Sequential hybridization: it is the most popular type; it performs different research methods sequentially so that the result of the first one serves as an initial solution for the next one.

- Synchronous parallel hybridization: this hybridization is carried out by incorporating a particular research method in an operator. It is more complex to

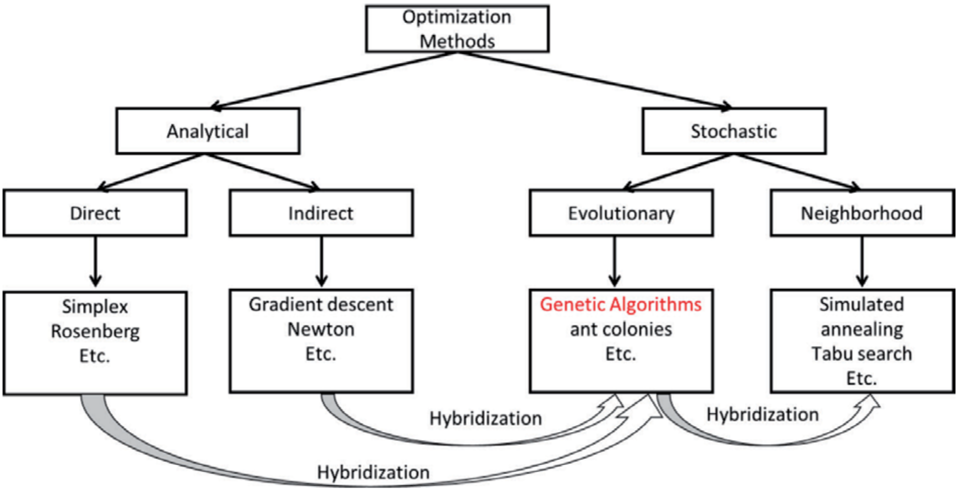


Figure 2.
Possible hybridization schemes.

implement than the previous one. The aim is to combine a LS with a global search for the purpose of improving convergence.

- Asynchronous parallel hybridization: the hybrid methods belonging to this class are characterized by an architecture such that two algorithms A and B are involved simultaneously, and each one adjusts the other. The Algorithms A and B share and exchange information throughout the research process.

A global view of the different possible hybridization schemes is shown in **Figure 2** where we can see hybridization between stochastic (particularly metaheuristics) and analytical optimization methods. Indeed, the common point between all these hybridizations is that they try to merge the strengths and eliminate the weaknesses of the different concepts. Therefore, the efficiency of the search solution space can still be improved, and new opportunities emerged which may lead to even more powerful and flexible research methods.

3. Genetic algorithm

GAs use a vocabulary inspired from natural genetics. However, the natural processes to which they refer is more complex [12]. They are iterative optimization procedures that repeatedly apply GA operators (such as selection, crossover, and mutation) to a group of solutions until some criterion of convergence has been satisfied. Note that the essential element of the genetic vocabulary is the "individual" that belongs to a set of individuals called "population". The individual is made up of a "chromosome" which is itself made up of "genes" which contain the inheritance characteristics of the individual. The principles of "selection", "crossover" and "mutation" are the main genetic operators that are inspired by the same natural processes. First, the selection operator includes generally two variants; the parental selection devoted for reproduction and the replacement selection devoted to keeping the population size constant by individuals who will survive in the next generation. A selection favors the best individuals but has also to give a chance to the less good to avoid the problems of premature convergence. Note that the "fitness" or "evaluation" function characterizing how well adapted the solution is to its environment, is closely related to the selection process. Then, the crossover operator involves combining the information from two parents to create one or two new individuals (or "offspring") making it possible to diversify the population and explore more the research space. At last, the mutation operator causes a small perturbation on an individual's chromosome thus allowing to guarantee gene diversity so that the algorithm does not get stuck in local minima. This process continues, throughout the generations, until the stop criteria is reached. The latter can be in various forms such as for example several global iterations defined in advance or the difference between the average population's fitness and the best one.

It is also important to underline that during the implementation of the algorithm it is necessary to take into account the encoding of the different parameters. In fact, one of the characteristics of GAs is their need to a genetic representation of the desired problem using an encoding. Therefore, the algorithm can evaluate the fitness of every individual and computes the problem's objectives and constraints based only on the encoding without any further knowledge of real parameters.

Figure 3 shows the optimization flowchart offered by GA whereas Reference [13] provides further discussion on the theoretical properties of GAs and on standard genetic operators.

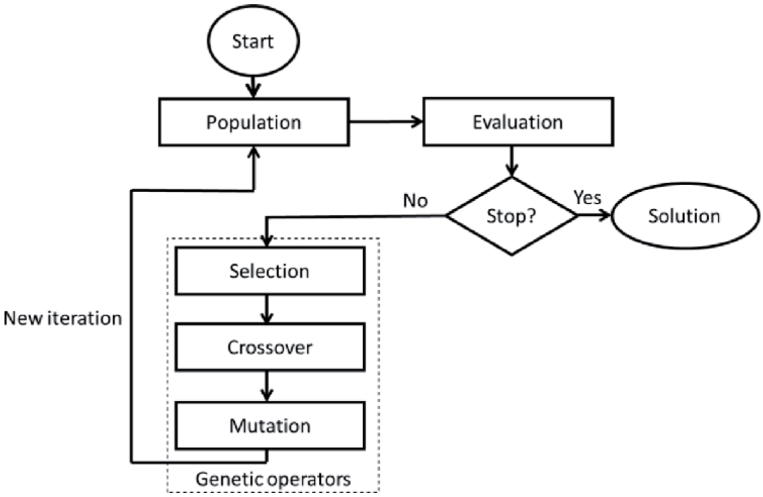


Figure 3.
Flowchart of the optimization procedure based on a genetic algorithm.

4. Hybrid genetic algorithm

The most natural hybridization in the literature is the use of a single solution meta-heuristic algorithm embedded in a population-based meta-heuristic algorithm like the unmissable GA in order to improve the qualities of the solutions. In fact, whereas population-based meta-heuristics tend to diversify the search space by exploring it different parts, LS meta-heuristics tend to intensify the search by exploiting only one part of the space.

LS methods are based on a neighborhood relation and on a procedure exploiting this neighborhood. They consist in moving from one solution to another closer in the space of the candidate solutions, until a solution considered as optimal is found, or the allowed time is exceeded. We can therefore summarize the main idea of a LS as follows:

- Start from an initial solution,
- Improve that solution,
- Repeat the process several times.

Hence, through the hybridization, individuals attempt to identify promising areas of the solution space which are then explored in more detail by LS methods. The best known among the latters are the simulated annealing, the tabu search or simply descent methods (iterated local searches).

Concerning the hybridization adopting the global GA, as its decision-making mechanics are based on random principles, this detailed research can be difficult to carry out. Heuristics therefore serve to fill this weakness. In a general way, at all generations, an individual (or some individuals) are randomly selected among the best in the population. Then, a set of solutions defining its (their) neighborhood is constructed. Afterwards, each of these neighbors is evaluated in order to check whether there is an improvement in the evaluation function. GA then manipulates solutions that have already been improved locally. This has the effect of allowing even more advantageous solutions to be generated more quickly. However, as there are several ways to hybridize GAs while maintaining a fairly modular program structure, a sequential approach could be advantageous in such a way that it is enough to let the GA run to a substantial level of convergence, then let the local optimization procedure follow up the process, for example by taking the 5% or the 10% best individuals of the last generation. However, two main approaches are also often followed: firstly, the Lamarckian approach [14] where direct learning transmits the best characteristics of each individual from generation to generation. Hence, both change in genotypic information and fitness are transmitted to the individual as genotypic information at the end of the LS, altering thereby the chromosomes. Secondly, the Baldwinian approach [14] where only the improved fitness function value is changed after the LS and not the genotypic information. The first approach is known to be faster than the second one but may cause premature convergence [13].

Note that usually the individuals that undergo LS are chosen uniformly at random, but it would be more efficient to use a tuning technique which consists of a primary conducted experiment aiming to find the optimal part of the population that should perform LS. This part is commonly referred to as the LS probability mentioned above that is used to run the actual experiment and remains fixed during the algorithm execution. In the literature, several techniques aim to reduce unnecessary local optimization and therefore additional calculation time, such as distribution-based techniques [15], fitness-based techniques [15] and LS potential [16] that have been proposed to select the optimal individuals among the given population that should do a LS. In the following, some main LS methods are remained.

4.1 Simulated annealing (SA)

Simulated annealing is an optimization technique inspired by the simulation methods of Metropolis (1950s) in statistical mechanics. It is distinguished by the introduction of a temperature parameter which is adjusted during research [17]. Moreover, in optimization, this method considers a neighborhood exploitation procedure which makes it possible to go towards a neighboring solution of less good quality with a non-zero probability, thus allowing to escape the local optima.

At the beginning, a "temperature" parameter T is determined and decreases throughout the algorithm to tend towards 0. The probability of accepting deteriorated solutions depends on the value of this parameter (the more the temperature T decreases, the more this probability decreases). The interest of simulated annealing lies in the fact that there is a proof of its asymptotic convergence. Thus, when certain conditions are verified (decrease diagram of T), we are guaranteed to obtain the optimal solution. However, the parameterization recommended by theory is not very realistic and it takes a long time to get to parameterize these methods. Note that this method may also require a stop criterion, if the "optimal" setting has not been found. Hybridization between GA and SA algorithms has been used in several domains [18–20].

4.2 Tabu search (TS)

Tabu search was introduced as a new strategy to escape local optima using a notion of memory [21]. Exploitation of the neighborhood makes it possible to move from the current solution to its best neighbor (this one not necessarily having a better quality than the current solution). To avoid cycling between a local optimum and its best neighbor, the method prohibits moving to a recently visited solution. To do this, a tabu list containing the attributes of the last visited solutions is kept up to date. Each new solution considered removes the oldest visited solution from this list. Thus, the search for the following current solution is done in the neighborhood of the current solution without considering the solutions belonging to the tabu list.

However, the size of this list is a method parameter that is difficult to adjust. In addition, this strategy requires the definition of a stopping criterion. We find the hybridization between GA and TS in many applications such as image processing [22], computational grids [23] and flow shop scheduling problem [24].

4.3 Iterated local search (ILS)

The descent methods are quick and simple to implement but generally do not lead to the best optima because they stop as soon as a local optimum is found [25]. In order not to get stuck on this local optimum, there are various strategies which allow the search to continue after having found an optimum. One strategy to overcome the abrupt stopping of the search for a local optimum is to iterate the descent method. The following steps are carried out from the found optimum:

- Apply a perturbation on the current solution,
- Apply a descent method on that solution.
- Chose via an acceptance criterion if the new optimum becomes the current solution and go back to the first step until the stop criterion is reached. Common stop criteria are the execution time, the number of iterations and the number of total evaluations.

The disturbance can consist in restarting a solution taken randomly in the search space or in choosing a solution in a neighborhood far from the optimum or even in choosing a neighbor of the same quality as the optimum.

Heuristic Iterative Local Search (ILS) [26] is based on a simple idea: instead of repeatedly applying a LS procedure from randomly generated solutions, ILS generates the starting solution for the next iteration by applying a perturbation on the local optimum found at the current iteration. This is done in the hope that the disturbance mechanism provides a solution located in the attraction basin of a better local optimum. Therefore, the disruption mechanism is a key part of ILS. In addition, an acceptance criterion defines the conditions that the new local optimum must meet in order to replace the current local optimum (**Figure 4**). Thus, the acceptance criterion, combined with the disruption mechanism, helps to control the trade-off between intensification and diversification. However, many acceptance

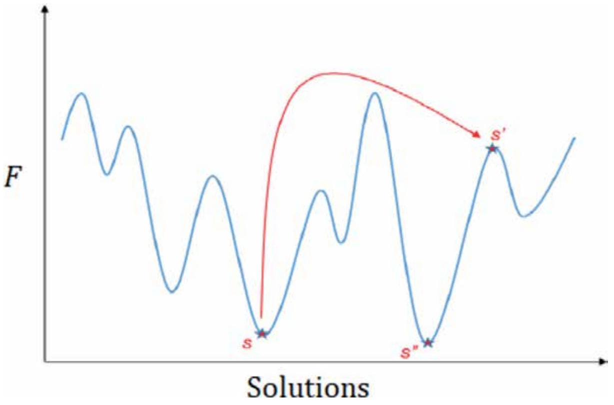


Figure 4.
From a local minimum s, a disturbance of the latter generates a minimum s' from which a LS is launched to arrive at a third local minimum s" potentially better than s.

criteria which reconcile the two objectives can be applied [12]. Hybridization between GA and ILS is one of the simplest and common hybridizations that can be found in several works [27–29].

4.4 Variable neighborhood search (VNS)

The Variable Neighborhood Search (VNS) is a meta-heuristic algorithm [30], based on the principle of systematic neighborhood change during LS. Indeed, at the initialization step, a set of neighborhood structures must be defined. These neighborhoods can be chosen arbitrarily, but often a sequence ordered in size growing neighborhoods is used. The VNS procedure consists of three stages: (1) the disturbance (shaking), (2) the LS and (3) the displacement. Recently, a variety of problems have involved hybridization between GA and VNS [31–33].

5. Test case

We consider here the following optimization problem: search for the minimum mass of an inter-stage skirt made of composite sandwich with Carbon Fiber Reinforced Polymer (CFRP) skins and aluminum Nida core (composites are fabricated by attaching two thin but stiff skins to a lightweight but thick core) that are located between the first stage and the second stage situated on the upper part of a linear launcher (**Figure 5**).

The structure is subjected to pressure oscillations from upstream solid propellant engines to ensure "payload comfort". The skirt therefore has a filtering role in order to limit the vibratory levels at the top of the launcher. To best dampen these oscillations, the structure must be mechanical sized according to a compromise between rigidity and flexibility linked to the need to hold in buckling, in addition to composite manufacturing constraints (mirror symmetry, balancing, grouping) [34]. Note that we need to optimize only one skin plus the core hence the final result is obtained by symmetry.

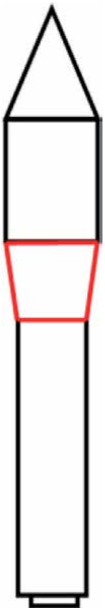


Figure 5.
Simplified model of the sandwich composite inter-stage skirt (in red) located on the upper part of the linear launcher.

Mathematically speaking, we can write the optimization problem as follows:

Objective: minimize the total mass (M).

Subject to:

$$\textbf{Constraints:} \begin{cases} \textit{structure's buckling}\,(B)(>2{,}03) \\ \textit{structure's stiffness}\,(S)(<2\mathrm{e}8\,\mathrm{N/m}) \\ \textit{fabrication constraints}\,(FC) \end{cases}$$

$$\textbf{Design variables:} \begin{cases} \textit{total number of plies forming the skin} \\ \textit{plies thicknesses} \end{cases}$$

5.1 Proposed hybrid genetic algorithm

To solve the above optimization problem, we have chosen the global GA. In fact, during the mechanical sizing of a composite structure, plies thicknesses are often predetermined, and plies orientations are usually restricted to a small set of angles due to manufacturing constraints and/or limited availability of experimental data. Most of the time, this gives rise mathematically to a discrete optimization problem. Hence GA is a suitable optimization method that we also decided to hybridize with a LS method to overcome the drawbacks of GA cited throughout this chapter.

First of all, an appropriate representation of the problem is recommended through the encoding of all the parameters. In fact, the initial population of the GA is composed of a set of skin plies (sandwich case) characterized by fiber orientations. Thus, an individual in the population is equivalent to a laminate (set of plies) and can simply be represented by an orientation chromosome (see **Figure 6**). Plies and core thicknesses of the structure are predefined based on composite design constraints [35] and are not considered herein as optimization variables. Therefore, an individual (a laminate) is represented by one chromosome regrouping one skin plies characterized by orientation angles (0°, 90°, ± 45°, ± 30°), encoded according to 5 possible values {0, 1, 2, 3, 4} called the chromosome domain where particularly add where 0° is encoded by 0, 90° is encoded by 1, ± 45° is encoded by 2, "no ply" is encoded by 3 and finally ± 30° is encoded by 4. "no ply" is encoded by 3.

To handle constraints, concerning the composite manufacturing ones, a repair strategy is adopted during the decoding phase (**Figure 7**). More explanation about these constraints handling method could be seen in [36]. Whereas mechanical constraints (buckling and stiffness) are managed through the genetic selection operator.

The proposed Hybrid Genetic Algorithm (HGA) herein is based on a strong and sequential cooperation between GA and a descent LS technique for the entire course

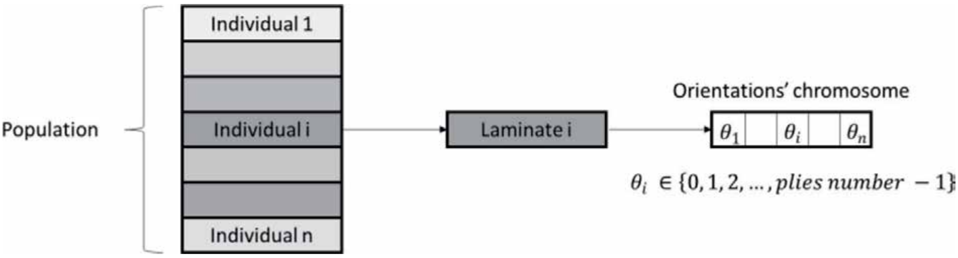
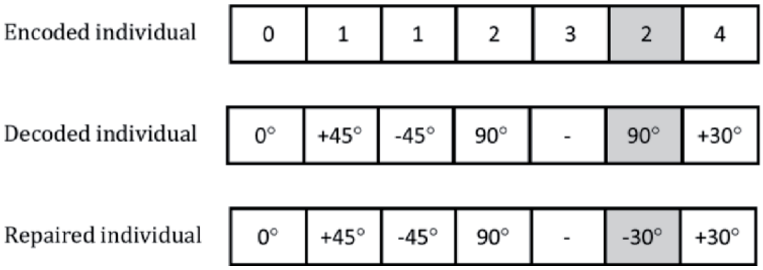


Figure 6.
The genetic representation of the design variables.

of the algorithm where the LS optimization is integrated at the beginning of the GA previous to the genetic operators to intensify research only around selected promising solutions. Hence, the adopted scheme is as follows: for every new iteration, the best individual reached by the GA is selected as an initialization for the LS method. Then, a set of two neighbors is built following a well-defined structure. The latter aims to verify if there is any neighboring orientation that allows an improvement of the individual's fitness by randomly selecting a gene related to the orientation and flipping it to either the previous or the next value in the concerned chromosome's domain while of course respecting all manufacturing constraints via the repair cited repair strategy. For example, if the selected gene's code is 2, the value of this gene will be replaced by 1 in the first neighbor and by 3 in the second one. However, if the randomly selected gene is the domain's upper bound (4) or the lower one, then the two neighbors are built by flipping firstly the gene to the previous value in the domain (3) then to the lower bound (0) or conversely.

Afterwards, each neighbor is evaluated and if one's fitness turns out to be better, then it is introduced in the population to replace the first individual and so on **Figure 7** shows an example of the decoding and repair stategy where the steps decoding then reparing are explained, the randomly selected gene's code is 2, corresponding to 90°. In order to respect the balancing composite rule, that gene should



Encoded individual	0	1	1	2	3	2	4
Decoded individual	0°	+45°	-45°	90°	-	90°	+30°
Repaired individual	0°	+45°	-45°	90°	-	-30°	+30°

Figure 7.
Example of the decoding and repair strategy.

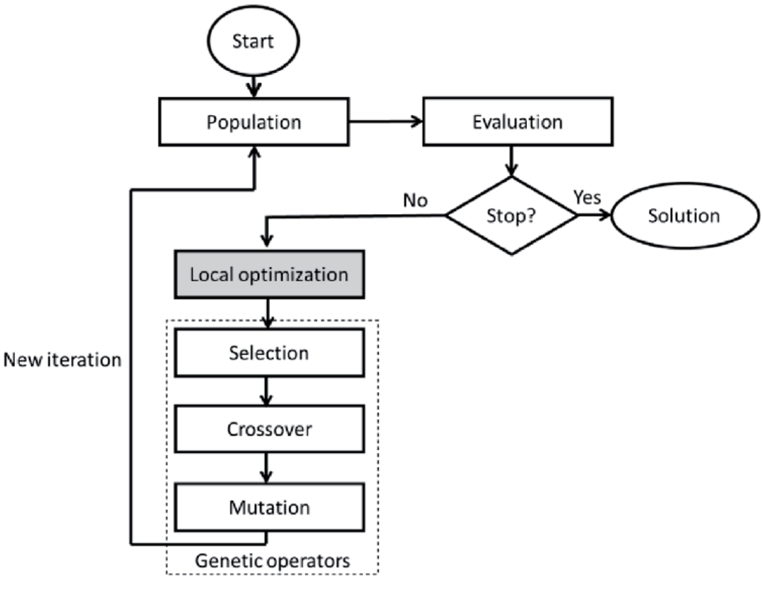


Figure 8.
Flowchart of the optimization procedure based on the hybrid genetic algorithm (HGA).

Parameters	Values
Total number of iterations	100
Number of individuals per generation	30
Chromosomes' size	8
Mutation probability	0,2
Mutant gene per individual	1
Crossover probability	1
Number of individuals chosen during the selection	1
Number of iterations for the LS	5
Number of neighbors per individual	2

Table 1.
Initialization of the algorithms' parameters.

be replaced by -30° (repair) so the entire laminate respects the composite design constraints. The **Figure 8** shows the adopted HGA flowchart.

5.2 Application and results

A comparative study between the classical GA and the proposed HGA is presented in the following. GA and HGA are implemented in a Python 3 environment and computed based on the same parameters (see **Table 1**). Note that at each iteration, the algorithms run through a database containing individuals already assessed with their respective fitness calculated through the finite element analysis program called NASTRAN [37] and look for the latter from the stacks previously evaluated. Besides, two criteria are implemented to stop the running of GA: the first stops the algorithm after a fixed number of generations have been created whereas the second stops after a given number of generations without improvement of the best individual in the population. This second criterion assures a certain quality of the best design, even if it is only relative to the previous best design. The comparison criterion is the fitness and the execution time associated with the number of iterations to achieve convergence. Concerning the HGA, the stop criteria is a prior fixed number of iterations.

Table 2 summarizes the optimization results. After 100 iterations, the best reached laminate is composed of 6 plies per skin against 8 initially and respects all the design constraints of composite structures thanks to the adopted decoding and repair strategy. Besides both stiffness and buckling constraints are satisfied with margins of +39.2% and + 5.9% for GA and HGA, respectively. Furthermore, the total mass is decreased by 8.4% compared to reference. This shows that generally all genetic processes perform well by offering a convincing solution. In fact,

	Laminate (°)	$S < 2^{e}8$ (N/m)	B > 2.03	M (Kg)	Convergence (Iterations/Time CPU)
NASTRAN	[45/0/30/0/−30/0/45/90]	$1.9^{e}8$	2.73	357	—
GA	[30/0/−30/30/−30/90]	$1.4^{e}8$	2.15	327	26/70 min
HGA	[30/0/−30/30/−30/90]	$1.4^{e}8$	2.15	327	18/54 min

Table 2.
Results of the comparative study between GA and HGA.

the program converges always towards the optimal solution. However, HGA provides a decrease from 26 to 18 in the number of needed iterations to converge with a gain of 24 minutes on the associated elapsed time. Thereby, HGA can outperform the standard GA on this sandwich composite structure optimization problem which highlight the hybridization mechanism and its capacity in the improvement of convergence.

6. Conclusion

In this chapter we have underlined the importance of the process of hybridization in ameliorating the global genetic algorithm behavior and especially its convergence. In fact, although genetic algorithms are effective complete search algorithms able to combine both exploration and exploitation with crossover and mutation operators, they suffer from being computationally expensive and can hence be improved using local search methods, so they can be made competitive with other algorithms when the search space is too large to explore. More generally, the combination of metaheuristics based on population as well as on local search, may offer more effective and dynamic methods for the solution of several problems.

The following important questions were asked: (1) when to apply local search optimization (2) to which individuals in the genetic algorithm should local searches be applied, and (3) what are the possible hybridizations that could improve the optimization results. Some answers to these questions have been provided particularly the reminder of the hybridization strategies and its different possible architectures in addition to the main local search methods known by their relevant coupling with the classical genetic algorithm. The latter being able to undergo different forms of hybridization thanks to its flexibility which is nevertheless one of its strongest points.

The application possibilities of these approaches are unlimited but in this chapter the test case that was presented is about an optimization algorithm combining traditional genetic operators with an iterated local search method specially designed to pursue the problem of mechanical sizing of a composite structure. The genetic algorithm was chosen as the global optimization tool because of its ability to deal with non-convex and discrete optimization problems, of which the design of laminated composites is an example. Besides, its flexible structure makes it possible to integrate the management of different mechanical and manufacturing constraints through genetic operators and the implementation of specific optimization processes such as the decoding and repair strategy. The developed hybrid algorithm was validated by comparing its results to those obtained via the classical genetic algorithm as well as a reference finite element analysis software. That test case showed hence that the developed hybrid genetic approach is able to obtain efficient results in a real-world problem related to the aerospace sector.

Acknowledgements

The author wishes to thank the French National Center for Space Studies that partially supported this work.

Conflict of interest

The authors declare no conflict of interest.

Author details

Leila Gharsalli
Polytechnic Institute of Advanced Sciences (IPSA), Ivry-sur-Seine, France

*Address all correspondence to: leila.gharsalli@ipsa.fr

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

References

[1] Holland JH. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology. Control and Artificial Intelligence. MI: University of Michigan Press; 1975

[2] Moscato PA. Memetic algorithms: A short introduction. In: Corne D, Dorigo M, Glover F, editors. New Ideas in Optimization. London: McGraw-Hill; pp. 219-234; 1999

[3] Chen X, Ong Y, Lim M, Tan KC. A multi-facet survey on memetic computation. IEEE Transactions on Evolutionary Computation. 2011;**15**(5):591-607. DOI: 10.1109/ TEVC.2011.2132725

[4] Wan W, Birch JB. An improved hybrid genetic algorithm with a new local search procedure. Journal of Applied Mathematics. Virginia Tech Publisher. Vol. 2013, p.10. Article ID 103591. DOI: 10.1155/2013/103591

[5] Bencheikh G, Boukachour J, El Hilali AA. A memetic algorithm to solve the dynamic multiple runway aircraft landing problem. Journal of King Saud University-Computer and Information Sciences. 2016;**28**(11):98-109

[6] Gharsalli L, Guérin Y. Mechanical sizing of a composite launcher structure by hybridizing a genetic algorithm with a local search method. Composites Part C: Open Access. 2021;**5**:100125. DOI: 10.1016/j.jcomc.2021.100125

[7] Montazeri M, Montazeri M, Reza Naji H, Faraahi A. A novel memetic feature selection algorithm. In: The 5th Conference on Information and Knowledge Technology. Shiraz, Iran: IEEE publisher; 2013. DOI: 10.1109/ IKT.2013.6620082

[8] Shyr WJ, Wang BW, Yeh YY, Su TJ. Design of optimal PID controllers using memetic algorithm. In: Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301). Anchorage, AK, USA. Vol. 3. 2002. pp. 2130-2131. DOI: 10.1109/ ACC.2002.1023951

[9] Labadi N, Prins C, Reghioui M. A memetic algorithm for the vehicle routing problem with time windows. RAIRO-Operations Research-Recherche Opérationnelle. 2008;**42**(3):415-431. DOI: 10.1051/ro:2008021

[10] Tse S-M, Liang Y, Leung K-S, Lee K-H, Mok T-S. A memetic algorithm for multiple-drug cancer chemotherapy schedule optimization. IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics. 2007;**37**(1):84-91. DOI: 10.1109/ tsmcb.2006.883265

[11] Talbi EG. A taxonomy of hybrid metaheuristics. Journal of Heuristics. 2002;**8**:541-564. DOI: 10.1023/ A:1016540724870

[12] Talbi EG. Metaheuristics: From Design to Implementation. Vol. 74. Hoboken, New Jersey: John Wiley & Sons; 2009. DOI: 10.1002/9780470496916

[13] Whitley D. A genetic algorithm tutorial. Statistics and Computing. 1994;**4**:65-85. DOI: 10.1007/ BF00175354

[14] Julstrom BA. Comparing Darwinian, Baldwinian, and Lamarckian search in a genetic algorithm for the 4-cycle problem. In: Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference. Orlando, Florida, USA: Morgan Kaufmann Publishers Inc; 1999. pp. 134-138

[15] Hart WE. Adaptive Global Optimization with Local Search [Thesis]. USA: University of California at San Diego; 1994

[16] Shannon Land MW, K. Belew R. Evolutionary Algorithms with Local Search for Combinatorial Optimization (thesis). CA, USA: University of California, San Diego; 1998

[17] Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. In: Fischler MA, Firschein O, editors. Readings in Computer Vision. San Francisco, CA, United States: Morgan Kaufmann publisher 1987. pp. 606-615. DOI: 10.1016/B978-0-08-051581- 6.50059-3

[18] Chen P-H, Shahandashti SM. Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints. Automation in Construction. 2009;**18**(14):434-443. DOI: 10.1016/j.autcon.2008.10.007

[19] Li Y, Guo H, Wang L, Fu J. A hybrid genetic-simulated annealing algorithm for the location-inventory-routing problem considering returns under E-supply chain environment. The Scientific World Journal. Vol. 2013. p.10. DOI: 10.1155/2013/125893

[20] Chen D, Lee C-Y, Park C-H. Hybrid genetic algorithm and simulated annealing (HGASA) in global function optimization. In: 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05). Hong Kong, China: IEEE publishe; 2005. pp. 5-133. DOI: 10.1109/ICTAI.2005.72

[21] Glover F, Laguna M. Tabu serach. In: Handbook of Combinatorial Optimization. Boston, MA: Springer; 1998. pp. 2093-2229

[22] Hadded M, Jarray F, Tlig G, Hasni H. Hybridization of genetic algorithms and tabu search for reconstructing convex binary images from discrete orthogonal projections. International Journal of Metaheuristics. 2014;**3**(4):291-319

[23] Xhafa F, Gonzalez JA, Dahal KP, Abraham A. A GA (TS) hybrid algorithm for scheduling in computational grids. In: Corchado E, Wu X, Oja E, Herrero Á, Baruque B, editors. Hybrid Artificial Intelligence Systems. HAIS 2009. Lecture Notes in Computer Science. Vol. 5572. Berlin, Heidelberg: Springer; 2009. DOI: 10.1007/978-3-642-02319-4_34

[24] Umam MS, Mustafid M, Suryono S. A hybrid genetic algorithm and tabu search for minimizing make span in flow shop scheduling problem. Journal of King Saud University-Computer and Information Sciences. Vol. 53. 2021. DOI: 10.1016/j.jksuci.2021.08.025

[25] Papadimitriou C, Steiglitz K. Combinatorial optimization: Algorithms and complexity. Mineola, New York: Dover Publications, Inc.; 1998

[26] Stützle T. Local search algorithms for combinatorial problems - analysis, improvements, and new applications. Darmstadt University of Technology, Germany, 1999. Vol. 220. pp. I-XI, 1-203. ISBN 978-3-89601-220-3

[27] Derbel H, Jarboui B, Hanafi S, Chabchoub H. Genetic algorithm with iterated local search for solving a location-routing problem. Expert Systems With Applications. 2012;**39**(13): 2865-2871. DOI: 10.1016/j.eswa.2011. 08.146

[28] Gharsalli L, Guérin Y. A hybrid genetic algorithm with local search approach for composite structures optimization. In: 8th European Conference for Aeronautics and Space Sciences. Madrid. AIP publisher. 2019. p. 9. DOI: 10.13009/ EUCASS2019-550

[29] Shuli H, Huan L, Xiaoli W, Ruizhi L, Junping Z, Jianan W. A hybrid framework combining genetic algorithm with iterated local search for the dominating tree problem. Mathematics. 2019;**7**(14):359. DOI: 10.3390/ math7040359

[30] Mladenovic N, Hansen P. Variable neighborhood search. Computers and Operations Research. 1997;**24**(111): 1097-1100. DOI: 10.1016/S0305-0548(97)00031-2

[31] Duda J. A hybrid genetic algorithm and variable neighborhood search for multi-family capacitated lot-sizing problem. Electronic Notes in Discrete Mathematics. 2017;**58**:103-110. DOI: 10.1016/j.endm.2017.03.014

[32] Fuksz L, Pop PC. A Hybrid genetic algorithm with variable neighborhood search approach to the number partitioning problem. In: Pan JS, Polycarpou MM, Woźniak M, de Carvalho ACPLF, Quintián H, Corchado E. (eds). Hybrid Artificial Intelligent Systems HAIS. Lecture Notes in Computer Science, vol. 8073. Berlin, Heidelberg: Springer; 2013. DOI: 10.1007/978-3-642-40846-5_65

[33] Luo Y, Pan Y, Li C, Tang H. A hybrid algorithm combining genetic algorithm and variable neighborhood search for process sequencing optimization of large-size problem. International Journal of Computer Integrated Manufacturing. 2020;**33**(110-11): 962-981. DOI: 10.1080/0951192X.2020.1780318

[34] Mårtensson P, Zenkert D, Åkermo M. Effects of manufacturing constraints on the cost and weight efficiency of integral and differential automotive composite structures. Composite Structures. 2015;**134**: 572-578. DOI: 10.1016/j.compstruct.2015.08.115

[35] Gay D. Composite Materials: Design and Applications. Third ed. Boca Raton, Floride, USA: CRC Press; 2014

[36] Puzzi S, Carpinteri A. A double-multiplicative dynamic penalty approach for constrained evolutionary optimization. Structural and Multidisciplinary Optimization. 2008;**35**:431-445. DOI: 10.1007/s00158-007-0143-1

[37] MSC Nastran 2012 Quick Reference Guide. MSC Software corporation. USA. 2012

Chapter 3

Flexible Project Scheduling Algorithms

Zsolt T. Kosztyán

Abstract

Despite the emerging importance of flexible project management approaches, such as agile extreme and hybrid methodologies, the algorithmic support of these approaches is still insufficient. In addition, single project scheduling has received far more attention than have schedules of multilevel projects, such as project portfolios or multi projects. This lack of scheduling techniques is especially true for flexible portfolios, such as agile, hybrid, and extreme project portfolios. While multilevel project scheduling algorithms already exist for fixed multilevel project structures, they are not able to handle flexible structures. This chapter proposes algorithms to schedule both flexible single and multilevel projects. The proposed algorithms handle both flexible and unplanned tasks and dependencies. They handle both single and multimode completion modes, and both renewable and nonrenewable resources. In addition, this chapter proposes a matrix-based risk-valuation framework to evaluate risk effects for flexible projects and portfolios. With this framework, project scheduling approaches are compared.

Keywords: agile, extreme, hybrid projects, schedule, multilevel projects, agents

1. Introduction

Despite the flexibility, such as agile (APMa), hybrid (HPMa), and extreme (EPMa), project management approaches come from the software project environment [1], and they are being increasingly used in nonsoftware environments as well. Flexible project management methods require flexible project scheduling methods, which allow flexible project structure [2]. Because of the time complexity, only a few methods handle the flexible nature of the projects. Nevertheless, in recent years, a new family of flexible scheduling methods [3] has been proposed. Instead of network-based methods, these algorithms are based on domain mapping (DMM) [4] and multidomain mapping matrices (MDMs) [5]. To support the agile project management approach, the proposed matrix-based methods handle both the priority of the task completions and the flexibility of dependencies between tasks [2, 6]. In addition to supporting extreme project management approaches, unplanned tasks can be scheduled [7]. The proposed matrix-based method can also be used to plan traditional, nonflexible projects. In this way, multimode completions can be specified. Nevertheless, the multimode and flexible project structures can also be combined to support hybrid project management approaches. The proposed flexible scheduling algorithms can be used not only in the projects but also at the project portfolio level [8].

The proposed matrix-based algorithm is based on the former studies [2, 6–8]; however, they are unified into the common matrix-based model, and they are extended to handle nonrenewable resources too. The proposed matrix-based model and the scheduling algorithm have the following features.

1. They handle fixed, but also *flexible* dependencies and mandatory but also *supplementary* tasks; therefore, traditional but *agile* project plans can also be modeled.

2. They handle planned, but also *unplanned* tasks, therefore, they can also model *extreme* project plans.

3. They handle single, but *multimode* completion modes; therefore they can also model traditional, but *hybrid* project plans, too.

4. They can handle *renewable*, but *non-renewable* resources, too.

5. They can model single, but also *multilevel* project plans.

In order to keep the adaptability of the proposed method, all features in italic style are optional. It means, when scheduling traditional project plans, there is no need to plan any supplementary tasks, flexible dependencies, or unplanned tasks; however, these features are optionally used in a flexible, such as agile, hybrid, or extreme project planning. Selecting from the multiple completion modes (called discrete technologies) is only relevant if there are more alternative technology. And last but not least, planning in a multilevel project environment is also an optional feature; however, it is crucial in the case of planning the resource sharing within the multi-projects.

In addition to the tasks, the risk effects of project scheduling can also be modeled [9]. The matrix-based risk evaluation has the following features:

1. Risk factors, risk effects, goals, and stakeholders can be modeled in a unified matrix-based model.

2. It handles independent, but also interdependent risk factors.

3. Both the planning and the tracking phases are covered.

All the proposed algorithms have software application support [10]. Therefore, after the model has been developed, the proposed methods can be compared. The MATLAB add-on [10] with examples guides users from the project planning to the risk evaluation.

The aim of this chapter is twofold. First, the chapter shows how to model flexible single and multilevel project plans and their risks. Second, the chapter summarizes the algorithms, so-called project management agents, that simulate a decision maker. In this way, the different project management approaches can compete, and the best one can be selected.

2. Matrix-based project planning models

In terms of scheduling, a project is a set of tasks, which has to be solved,

1. within a time-frame (= time constraint, C_t),

2. within the budget (= cost constraint, C_c),

3. within the renewable/nonrenewable resources $(\mathbf{C}_r, \mathbf{C}_n)$, and

4. with adequate quality (C_q),

5. with adequate scope (C_s).

The fixed dependency between (successor and a predecessor) tasks specifies, that a successor task may start if the predecessor task has finished (sequential completion). While flexible dependency, based on a later decision either allows either a sequential or a parallel completion.

In terms of planning and scheduling, the multilevel project is the set of projects. Within a multilevel project, at least two overlapping projects specify a *multiproject* [11], if they have common resources. While at least two projects specify *programs*, if the goals of the projects are similar and they have dependency between them. In terms of scheduling, *project portfolios* can contain single projects, multiprojects, and programs too. While, *multilevel projects* can contain project portfolios, too.

Apart from network planning methods, matrix-based project planning is used to model complex project plans [12]. Matrix-based project planning methods are often based on the design (or dependency) structure matrix (DSM) [13]. The domain mapping matrix (DMM) is an extended version of the DSM, with multiple domains [4]. In this chapter, a modified project-oriented version of a domain mapping matrix (DMM) is used, which is called the project domain matrix (PDM) [2].

The PDM contains two mandatory and four supplementary domains.

LD The logic domain is an n by n matrix, where n is the number of tasks. Each cell contains a value from the [0,1] interval.

TD The time domain is an n by m matrix with positive real values, where m is the number of completion modes[1].

The first mandatory domain is the logic domain. Diagonal values in **LD** represent the priority values of the tasks. If a diagonal value is 0, then this task will not be completed. If the diagonal value is 1, then the task is a mandatory task, while if the diagonal value is between 0 and 1, then it is a supplementary task, which means that depending on the decision, either it will be completed or omitted/postponed. Out-diagonal values represent the dependency between tasks. If an out-diagonal value $[\mathbf{LD}]_{ij} \in \mathbf{LD}$ is 1, then task i precedes task j. In the case of $[\mathbf{LD}]_{ij} = 0$, there is no precedence relation from task i to task j. If $0 < [\mathbf{LD}]_{ij} < 1$, then there is a flexible dependency between task i and task j, which means the dependency is on whether decision task i precedes task j. Since all project networks from the considered databases do not contain any cycle, in other words, they can be ordered topologically, the logic domain of the topologically ordered project networks is an upper triangular (sub) matrix. Formally $[\mathbf{LD}]_{ij} := 0$, if $i > j$. The other mandatory domain of the PDM is the time domain. The positive values of the time domains represent the possible duration of tasks. For each task, k duration values can be assigned; nevertheless, the duration values may also match each other.

The additional supplementary domains are:

CD Cost domain, which is an n by m nonnegative matrix of task costs

QD Quality domain, which is an n by m nonnegative matrix of quality parameters of tasks

[1] A task within a project can be solved by different kind of technology, which requires different kind of (time, cost, quality, resource) demands and it has different kind of quality parameters. These technologies are called as *completion modes*.

ND The nonrenewable resource domain is an n by $m \cdot \eta$ nonnegative matrix of nonrenewable resource demands, where η is the number of types of nonrenewable resources.

RD The renewable resource domain is an n by $m \cdot \rho$ nonnegative matrix of renewable resource demands, where ρ is the number of types of renewable resources.

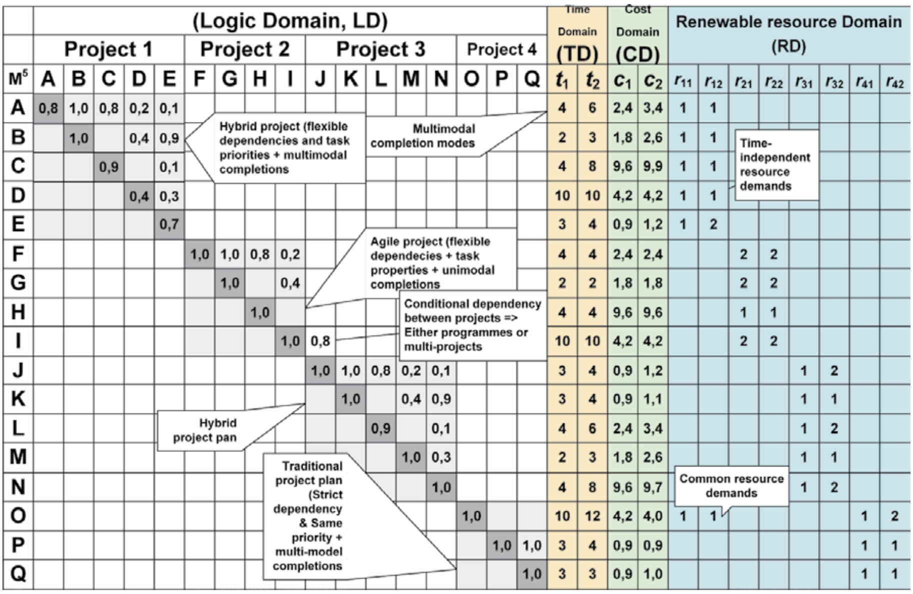
Table 1 shows an example of a fully filled PDM matrix. There are 3 (2 mandatory, 1 supplementary) tasks, 3 (2 fixed, 1 flexible) dependencies, 2 completion modes, 2 nonrenewable resources, and 3 renewable resources. The optional domains can be either ignored or filled out with zero values.

Since PDM can model flexible dependencies and task priorities, it can be used to model both traditional and flexible approaches, such as agile and extreme approaches, and hybrid project planning approaches (see details in Section 3). Nevertheless, handling completion priorities and flexible dependencies alone raises the number of possible project plans.

The project can be organized into a multilevel project. The projects in the applied M^5 (matrix-based multimode multilevel (project) management model) share their domains. **Table 2** shows an example of a multilevel project plan. The common logic domain allows us to plan flexible dependencies both within and between projects. It

PDM	Logic domain			Time domain		Cost domain		Quality domain		Nonrenewable resource domain				Renewable resource domain					
	A	B	C	t_1	t_2	c_1	c_2	q_1	q_2	n_{11}	n_{12}	n_{21}	n_{22}	r_{11}	r_{12}	r_{21}	r_{22}	r_{31}	r_{31}
A	1	1	.6	1	2	3	2	.7	.8	4	5	3	6	3	6	3	5	4	2
B	0	.7	1	2	2	4	3	.8	.9	3	3	4	4	4	7	4	6	4	3
C	0	0	1	4	3	5	3	.9	.8	4	2	5	3	4	2	4	7	6	4

Table 1.
The structure of the project domain matrix.



	(Logic Domain, LD)																	Time Domain (TD)		Cost Domain (CD)		Renewable resource Domain (RD)							
	Project 1					Project 2				Project 3					Project 4														
M^5	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	t_1	t_2	c_1	c_2	r_{11}	r_{12}	r_{21}	r_{22}	r_{31}	r_{32}	r_{41}	r_{42}
A	0,8	1,0	0,8	0,2	0,1													4	6	2,4	3,4	1	1						
B		1,0		0,4	0,9													2	3	1,8	2,6	1	1						
C			0,9		0,1													4	8	9,6	9,9	1	1						
D				0,4	0,3													10	10	4,2	4,2	1	1						
E					0,7													3	4	0,9	1,2	1	2						
F						1,0	1,0	0,8	0,2									4	4	2,4	2,4			2	2				
G							1,0		0,4									2	2	1,8	1,8			2	2				
H								1,0										4	4	9,6	9,6			1	1				
I									1,0	0,8								10	10	4,2	4,2			2	2				
J										1,0	1,0	0,8	0,2	0,1				3	4	0,9	1,2					1	2		
K											1,0		0,4	0,9				3	4	0,9	1,1					1	1		
L												0,9		0,1				4	6	2,4	3,4					1	2		
M													1,0	0,3				2	3	1,8	2,6					1	1		
N														1,0				4	8	9,6	9,7					1	2		
O															1,0			10	12	4,2	4,0	1	1					1	2
P																1,0	1,0	3	4	0,9	0,9							1	1
Q																	1,0	3	3	0,9	1,0							1	1

Hybrid project (flexible dependencies and task priorities + multimodal completions

Multimodal completion modes

Time-independent resource demands

Agile project (flexible dependecies + task properties + unimodal completions

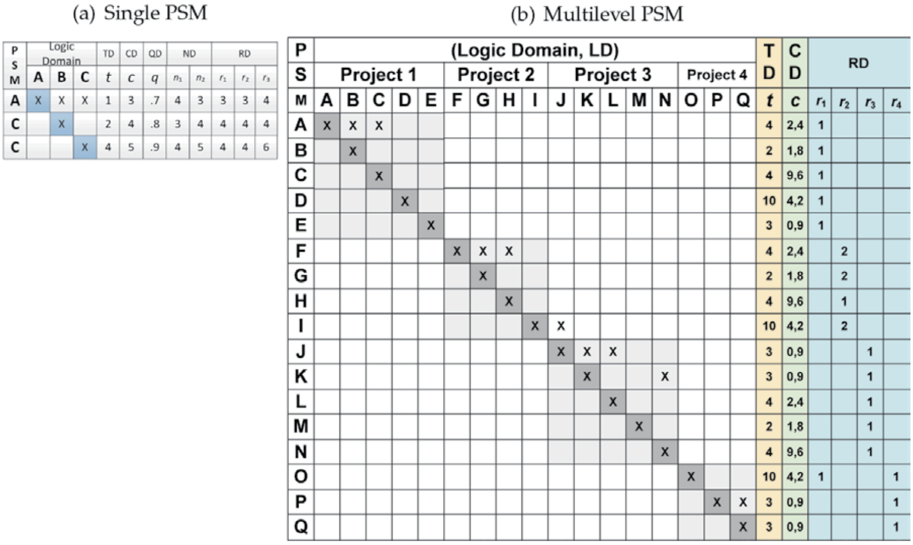
Conditional dependency between projects => Either programmes or multi-projects

Hybrid project pan

Traditional project plan (Strict dependency & Same priority + multi-model completions

Common resource demands

Table 2.
Matrix-based multimode multilevel (project) management model.



(a) Single PSM

P S M	Logic Domain			TD	CD	QD	ND		RD		
	A	B	C	t	c	q	n_1	n_2	r_1	r_2	r_3
A	X	X	X	1	3	.7	4	3	3	3	4
C		X		2	4	.8	3	4	4	4	4
C			X	4	5	.9	4	5	4	4	6

(b) Multilevel PSM

P	(Logic Domain, LD)																	T	C	RD			
S	Project 1					Project 2				Project 3					Project 4			D	D				
M	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	t	c	r_1	r_2	r_3	r_4
A	X	X	X															4	2,4	1			
B		X																2	1,8	1			
C			X															4	9,6	1			
D				X														10	4,2	1			
E					X													3	0,9	1			
F						X	X	X										4	2,4		2		
G							X											2	1,8		2		
H								X										4	9,6		1		
I									X	X								10	4,2		2		
J										X	X	X						3	0,9			1	
K											X			X				3	0,9			1	
L												X						4	2,4			1	
M													X					2	1,8			1	
N														X				4	9,6			1	
O															X			10	4,2	1			1
P																X	X	3	0,9				1
Q																	X	3	0,9				1

Table 3.
Single and multilevel PSM.

handles the different completion modes; therefore, all the traditional, hybrid, and agile project plans can be planned (see **Table 2**).

Because of the numerous possible project plans, there is no chance to compute all possible projects or multilevel project plans. A fast, exact method is required to find the best project or multilevel project structure.

After deciding which tasks are completed, which flexible dependency is required, and which completion mode is selected, we obtain a (multilevel) project plan. **Table 3** shows, a single (a) and a multilevel (b) project schedule matrix (PSM).

Table 3(a) shows a project schedule matrix of PDM (see **Table 1**), which contains 6 domains. **Table 3(b)** shows a possible project schedule M^5 matrix representation of a hybrid multilevel project (see **Table 2**), where there are 4 domains. In both models, empty cells represent 0 values, and 'X' represents 1 value. 'X' also indicates that this value is the result of a decision. The decision is always binary: either include or exclude a task or a dependency. The PSM contains only one mode, i.e., the selected completion mode for every task. PSM optionally contains the scheduled starts (SS) of tasks. Otherwise, the default start is the early start (ES) of tasks.

3. Project management agents

Previously, [1] found that in his study of the practices of software project managers, only 20% of IT projects were managed by a traditional project management (TPM) methodology. Generally, methods for investment and construction projects cannot be directly applied to software development or R&D projects, as these are managed by agile project management (APM) approaches. Currently, hybrid (i.e., combinations of traditional and agile and extreme) approaches are becoming increasingly popular [14]. However, flexible approaches are thus far not privileges for software development projects [15]. Rapidly changing environments increasingly enforce flexible approaches. Project planning and scheduling algorithms can support decision makers in managing projects; however, there are only a few algorithmic procedures that can support flexible approaches. Therefore, it is important to study how to extend project planning and scheduling methods to handle flexible and changing environments. Planning and scheduling methods, as agents, can also imitate decision makers; therefore, not only the methods but also the scheduling and project planning approaches can be modeled (see **Table 4**).

While, a project manager who follows a traditional project management (TPM) approach can use tradeoff or multimode methods to reduce task duration or cost/resource demands, an agile and extreme project manager tries to restructure the project. If the project structure is flexible (see **Table 4**), then the project duration can be reduced without increasing the project cost by reducing the number of flexible dependencies. In addition, in real project situations, decision makers can choose from different kinds of technologies (i.e. completion modes); therefore, the TPM and APM approaches can be integrated. Agile approaches usually split the projects into smaller so-called "sprints" that are usually 2–6 weeks. The content of sprints is specified by the customer and developers together. However, when running a sprint, unplanned new tasks and new requirements can be involved only until the next sprint. The extreme project management (EPM) approach handles the new tasks and new requirements during the implementation of the project. Extreme project management can confirm the extra costs and the increased project duration due to the extra tasks.

Flexible approaches require flexible project structures; however, in addition to the opportunity to reorganize the project, different kinds of technology (completion modes) should also be considered; therefore, traditional and flexible approaches should be combined into hybrid project management approaches [14–16]. Nevertheless, hybrid approaches should be supported by algorithmic methods to help decision makers ensure the project's success.

There are different combinations of agile and traditional project management approaches [14–16]. However, there are very few exact algorithms (see, e.g., [6, 7]) that can be used to solve hybrid multimode problems that can handle unplanned tasks and dependencies. Nevertheless, R&D and IT projects, such as introducing and setting up new information systems, may require reorganizing part of the project, and R&D projects may require handling unplanned tasks, particularly in the development phase. However, decreasing the time demands of mandatory tasks and those of the new unplanned tasks may also be an important requirement. Neither the agile approach, nor the extreme approach can handle this situation properly, nor can traditional approaches. Traditional approaches, or network-based methods, assume static logic plans, but the reorganization of projects may produce insufficient reductions in project duration and/or supplementary tasks, and important tasks may be excluded from the project due to budget constraints and/or project deadlines. A hybrid project management (HPM) approach can combine traditional, agile, and extreme approaches; however, these kinds of HPM

Planning approaches	Features			
	Project structure	New tasks	Multimode	Constraints
Traditional (TPM)	Fixed	Not allowed	Handled	Fixed
Agile (APM)	Flexible	Not allowed	Not handled	Fixed
Extreme (EPM)	Flexible	Allowed	Not handled	Flexible
Hybrid (HPM)	Flexible	Allowed	Handled	Optional

Table 4.
Comparison of the traditional and flexible approaches.

approaches are not yet supported by project planning methods. The proposed algorithm combines agile, extreme, and traditional approaches. This method extends the traditional multi-mode resource-constrained project scheduling problem by allowing for the restructuring and reorganizing of projects and handling of unplanned new tasks.

The proposed hybrid time–cost and hybrid time-quality-cost tradeoff models [6] and multimode methods [7] manage flexible project plans and allow us to restructure or reorganize these project plans to satisfy customer and management claims. In contrast to the traditional project scoring and selection methods, there is no need to specify all project alternatives to select the most desirable project scenario or the one with the shortest duration or lowest cost. The following definition specifies a matrix representation of a flexible (multilevel) project plan and its possible realization.

Definition 1. Denotes $\{\cdot\}$ as a supplementary domain, which can be an empty matrix. Denote $\mathbf{M} = [\mathbf{LD}, \mathbf{TD}, \{\mathbf{CD}\}, \{\mathbf{QD}\}, \{\mathbf{ND}\}, \{\mathbf{RD}\}]$ as a flexible (multilevel) project plan, where $\mathbf{LD}$ and $\mathbf{TD}$ are mandatory but $\mathbf{CD}, \mathbf{QD}, \mathbf{ND}, \mathbf{RD}$ are supplementary domains. $\mathbf{M}'' = [\mathbf{LD}'', \mathbf{TD}'', \{\mathbf{CD}''\}, \{\mathbf{QD}''\}, \{\mathbf{ND}''\}, \{\mathbf{RD}''\}]$ is a realized (fixed) (multilevel) project plan of $\mathbf{M}$ if $\mathbf{M}''$ is at least an n by $n+1$ but maximum n by $n+3+\eta+\rho$ matrix with two mandatory and six supplementary domains. The following properties are satisfied: if m denotes the number of nodes, ρ denotes the number of renewable resources, η denotes the number of nonrenewable resources, and p denotes the number of projects, then.

LD″: n by n logic domain, where $\mathcal{L}''_{i_k,j_s} = [\mathbf{LD}'']_{i_k,j_s} \in \{0,1\}$.. $\mathcal{L}''_{i_k,j_s} = \mathcal{L}_{i_k,j_s} = [\mathbf{LD}]_{i_k,j_s}$, if $\mathcal{L}_{i_k,j_s} \in \{0,1\}$ and either $\mathcal{L}''_{i_k,j_s} = 1$ or $\mathcal{L}''_{i_k,j_s} = 0$, if $0 < \mathcal{L}_{i_k,j_s} < 1, k, j = 1, .., p$.

TD″: n by 1 column vector (time domain), where $\mathcal{T}''_i = [\mathbf{TD}'']_{i_k} = \mathcal{T}_{i_k,\omega_{i_k}} = [\mathbf{TD}]_{i_k,\omega_{i_k}}$, and $i_k = 1, 2, .., n_k, \omega_{i_k} \in \{1, 2, .., m\}, k = 1, 2, .., p$.

CD″: n by 1 column vector (cost domain), where $\mathcal{C}''_i = [\mathbf{CD}'']_{i_k} = \mathcal{C}_{i_k,\omega_{i_k}} = [\mathbf{CD}]_{i_k,\omega_{i_k}}$, and $i_k = 1, 2, .., n_k, \omega_{i_k} \in \{1, 2, .., m\}, k = 1, 2, .., p$.

QD″: n by 1 column vector (quality domain), where $Q''_i = [\mathbf{QD}'']_{i_k} = \mathcal{Q}_{i_k,\omega_{i_k}} = [\mathbf{QD}'']_{i_k,\omega_{i_k}}$, and $i_k = 1, 2, .., n_k, \omega_{i_k} \in \{1, 2, .., m\}, k = 1, 2, .., p$.

ND″: n by η nonrenewable resource domain, where $\mathcal{N}''_{i_k,w} = [\mathbf{RD}'']_{i_k,w} = \mathcal{N}_{i_k,m\cdot(w-1)+\omega_{i_k}} = [\mathbf{ND}]_{i_k,m\cdot(w-1)+\omega_{i_k}}$, and $i_k = 1, 2, .., n_k, \omega_i \in 1, 2, .., m, k = 1, 2, .., p, w \in \{1, .., \eta\}$.

RD″: n by ρ renewable resource domain, where $\mathscr{R}''_{i_k,r} = [\mathbf{RD}'']_{i_k,r} = \mathscr{R}_{i_k,m\cdot(r-1)+\omega_{i_k}} = [\mathbf{RD}]_{i_k,m\cdot(r-1)+\omega_{i_k}}$, and $i_k = 1, 2, .., n_k, \omega_i \in 1, 2, .., m, k = 1, 2, .., p, r \in \{1, .., \rho\}$.

Definition 1 proposes a unified matrix-based model, both for single and multilevel project plans, and both for single and multimode completions. In addition, by increasing n allows involving the unplanned tasks.

3.1 Demands

Definition 2. Let $\mathbf{M}'' = [\mathbf{LD}'', \mathbf{TD}'', \{\mathbf{CD}''\}, \{\mathbf{QD}''\}, \{\mathbf{ND}''\}, \{\mathbf{RD}''\}]$ be a matrix representation of the realized (multilevel) project, which contains $p > 1$ projects. Assume that $i_k < j_k \Rightarrow [\mathbf{LD}'']_{i_k,j_k} = 0$. Denote $SF_{i_k} = SS_{i_k} + \mathcal{T}_{i_k}$ as the scheduled finish time of task i_k from project k, where SS_{i_k} is the scheduled start time and t_{i_k} is the duration of task i from project k. Denote $R_{r,k}(\tau)$ as the maximum resource demand of renewable resource r for project k at a time τ. Denote τ_{0_k} as the start time of project k and τ_0 as the start time of the multilevel project. The total project values are defined as follows:

TPT: Total project time $TPT_k\mathbf{M}''$) of project k and the duration of the multilevel project ($TPT\mathbf{M}''$).

$$TPT_k\mathbf{M}'' = \max_{i_k} SF_{i_k}, \tag{1}$$

$$TPT\mathbf{M}'' = \max_{i} SF_{i}. \tag{2}$$

TPC: Total project cost ($TPC_k\mathbf{M}''$) of project k and the total cost of the multilevel project ($TPC\mathbf{M}''$).

$$TPC_k\mathbf{M}'' = \sum_{i_k}[\mathbf{CD}'']_{i_k}, \tag{3}$$

$$TPC\mathbf{M}'' = \sum_{k} TPC_k\mathbf{M}'' \tag{4}$$

TPQ: (relative) total project quality ($TPQ_k\mathbf{M}''$) of project k and the total quality of the multilevel project ($TPQ\mathbf{M}''$).

$$TPQ_k\mathbf{M}'' = \sum_{i_k}[\mathbf{QD}'']_{i_k} / \max_{m}\, [\mathbf{QD}]_{i_{m,k}}, \tag{5}$$

$$TPQ\mathbf{M}'' = \sum_{k} TPQ_k\mathbf{M}'' / \max_{m}\, [\mathbf{QD}]_{i_{m,k}} \tag{6}$$

where m is the completion mode.

TPN: Total project nonrenewable resource demands j ($TPN_{e,k}\mathbf{M}''$) of project k and the total nonrenewable resource e of the multilevel project ($TPN_e\mathbf{M}''$).

$$TPN_{e,k}\mathbf{M}'' = \sum_{i_k}[\mathbf{ND}'']_{i_k}, \tag{7}$$

$$TPN_e\mathbf{M}'' = \sum_{k} TPN_{e,k}\mathbf{M}'' \tag{8}$$

TPR: Total project renewable resources ($TPR_{r,k}\mathbf{M}''$) of project k and the total project resources of the multilevel project for resource r ($TPR_r\mathbf{M}''$).

$$TPR_{r,k}\mathbf{M}'' = \max_{\tau_{0_k} \le \tau \le TPT_k\mathbf{M}''} R_{r,k}\tau, \tag{9}$$

$$TPR_r\mathbf{M}'' = \max_{\tau_0 \le \tau \le TPT\mathbf{M}''} R_r\tau \tag{10}$$

TPS: Total project score ($TPS_k\mathbf{M}, \mathbf{M}''$) of project k and the total score of the multilevel project ($TPS\mathbf{M}, \mathbf{M}''$).

$$TPS_k\mathbf{M}, \mathbf{M}'' = \sum_{l''_{i_k,i_k}=1} l_{i_k,i_k}, \tag{11}$$

$$TPS\mathbf{M}, \mathbf{M}'' = \sum_{l''_{i,i}=1} l_{i,i}. \tag{12}$$

3.2 Relative constraints

Definition 3. Denote $C_X, X \in \{T, C, Q, N, R, S\}$ as the time, cost, quality, nonrenewable resource, renewable resource, and score constraints, respectively. Denote $C_X\% = \frac{C_X - TPX_{\min}}{TPX_{\max} - TPX_{\min}}$ as the relative constraints.

It is important to note that the relative constraint should be within the [0,1] interval ($C_X\% \in [0,1]$) to find a feasible solution. Nevertheless, the minimal and maximal demands ($TPX_{\max}, TPX_{\min}$) can be calculated without specifying all possible solutions (see Section 3.4).

(C_{X_k}) constraints can be defined not only for project $k = 1, .., p$ but also for multilevel projects.

3.3 Target function

Either simple or composite target functions can be specified both for single and multilevel projects.

Simple target functions:

$$TPT_k \rightarrow \min, TPT \rightarrow \min. \tag{13}$$

$$TPC_k \rightarrow \min, TPC \rightarrow \min. \tag{14}$$

$$TPQ_k \rightarrow \max, TPQ \rightarrow \max. \tag{15}$$

$$TPN_{e,k} \rightarrow \min, TPN_e \rightarrow \min. \tag{16}$$

$$TPN_{r,k} \rightarrow \min, TPN_{\mathrm{r}} \rightarrow \min. \tag{17}$$

$$TPS_k \rightarrow \max, TPS \rightarrow \max. \tag{18}$$

where $k = 1, .., p, e = 1, .., \eta, r = 1, .., \rho$.

The composite target function handles all possible targets with their importance:

$$\prod_{X \in \{T,C,N,R\}} \left(\frac{TPX - TPX_{\min}}{TPX_{\max} - TPX_{\min}}\right)^{v_X} \cdot \prod_{X \in \{Q,S\}} \left(\frac{TPX_{\max} - TPX}{TPX_{\max} - TPX_{\min}}\right)^{v_X} \rightarrow \min \tag{19}$$

where v_X is the weight of the importance of demands ($\sum_X v_X := 1$).

3.4 Main properties of the exact evaluation

Due to the size constraints, only the main feature of the proposed algorithm is summarized. See the details in [6, 8].

1. The evaluation contains three steps.

   a. First, the diagonal values of the **LD** are evaluated. All supplementary tasks ($0 < [\mathbf{LD}]_{ii} < 1$) are decided either to exclude from or to include in the (multilevel) project.

   b. In the second step, all flexible tasks ($0 < [\mathbf{LD}]_{ij} < 1, i > j$) are evaluated. They are either excluded or included.

   c. In the third step, all completion modes are evaluated. For every task, a completion mode is selected.

2. For every kind of multilevel project plan, the minimal (maximal) demands can be specified without calculating all possible solutions.

a. If all supplementary tasks are excluded (included), the *TPS* is minimal (maximal).

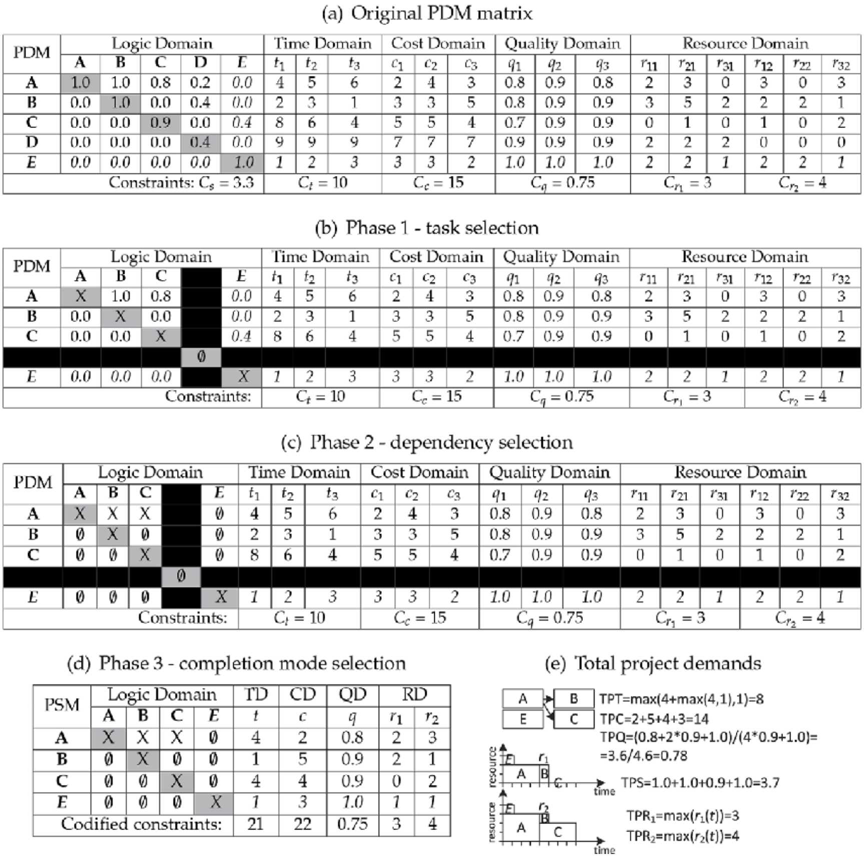
b. If all supplementary tasks are excluded (included), and all completion modes require minimal (maximal) demands, *TPC* and *TPN* are minimal (maximal).

c. If all supplementary tasks are excluded (included), and all dependencies are excluded (included) and all completion modes require minimal (maximal) demands, *TPT* is minimal (maximal).

d. If all supplementary tasks are excluded (included), but all dependencies are included (excluded) and all completion modes require minimal (maximal) demands, *TPR* is minimal (maximal).

3. Due to the evaluation, if the minimal (maximal) demands are greater (lower) than the constraint, neither the project plan nor their derived plans are feasible.

Based on these properties, exact back and forth algorithms are proposed to find a single project schedule [7] or multilevel project structure [8]. All methods contain



(a) Original PDM matrix

PDM	Logic Domain					Time Domain			Cost Domain			Quality Domain			Resource Domain					
	A	B	C	D	*E*	t_1	t_2	t_3	c_1	c_2	c_3	q_1	q_2	q_3	r_{11}	r_{21}	r_{31}	r_{12}	r_{22}	r_{32}
A	1.0	1.0	0.8	0.2	*0.0*	4	5	6	2	4	3	0.8	0.9	0.8	2	3	0	3	0	3
B	0.0	1.0	0.0	0.4	*0.0*	2	3	1	3	3	5	0.8	0.9	0.9	3	5	2	2	2	1
C	0.0	0.0	0.9	0.0	*0.4*	8	6	4	5	5	4	0.7	0.9	0.9	0	1	0	1	0	2
D	0.0	0.0	0.0	0.4	*0.0*	9	9	9	7	7	7	0.9	0.9	0.9	2	2	2	0	0	0
E	*0.0*	*0.0*	*0.0*	*0.0*	*1.0*	*1*	*2*	*3*	*3*	*3*	*2*	*1.0*	*1.0*	*1.0*	*2*	*2*	*1*	*2*	*2*	*1*
Constraints: $C_s = 3.3$						$C_t = 10$			$C_c = 15$			$C_q = 0.75$			$C_{r_1} = 3$			$C_{r_2} = 4$		

(b) Phase 1 - task selection

PDM	Logic Domain					Time Domain			Cost Domain			Quality Domain			Resource Domain					
	A	B	C		*E*	t_1	t_2	t_3	c_1	c_2	c_3	q_1	q_2	q_3	r_{11}	r_{21}	r_{31}	r_{12}	r_{22}	r_{32}
A	X	1.0	0.8		*0.0*	4	5	6	2	4	3	0.8	0.9	0.8	2	3	0	3	0	3
B	0.0	X	0.0		*0.0*	2	3	1	3	3	5	0.8	0.9	0.9	3	5	2	2	2	1
C	0.0	0.0	X		*0.4*	8	6	4	5	5	4	0.7	0.9	0.9	0	1	0	1	0	2
				∅																
E	*0.0*	*0.0*	*0.0*		X	*1*	*2*	*3*	*3*	*3*	*2*	*1.0*	*1.0*	*1.0*	*2*	*2*	*1*	*2*	*2*	*1*
Constraints:						$C_t = 10$			$C_c = 15$			$C_q = 0.75$			$C_{r_1} = 3$			$C_{r_2} = 4$		

(c) Phase 2 - dependency selection

PDM	Logic Domain					Time Domain			Cost Domain			Quality Domain			Resource Domain					
	A	B	C		*E*	t_1	t_2	t_3	c_1	c_2	c_3	q_1	q_2	q_3	r_{11}	r_{21}	r_{31}	r_{12}	r_{22}	r_{32}
A	X	X	X		∅	4	5	6	2	4	3	0.8	0.9	0.8	2	3	0	3	0	3
B	∅	X	∅		∅	2	3	1	3	3	5	0.8	0.9	0.9	3	5	2	2	2	1
C	∅	∅	X		∅	8	6	4	5	5	4	0.7	0.9	0.9	0	1	0	1	0	2
				∅																
E	∅	∅	∅		X	*1*	*2*	*3*	*3*	*3*	*2*	*1.0*	*1.0*	*1.0*	*2*	*2*	*1*	*2*	*2*	*1*
Constraints:						$C_t = 10$			$C_c = 15$			$C_q = 0.75$			$C_{r_1} = 3$			$C_{r_2} = 4$		

(d) Phase 3 - completion mode selection

PSM	Logic Domain				TD	CD	QD	RD	
	A	B	C	*E*	t	c	q	r_1	r_2
A	X	X	X	∅	4	2	0.8	2	3
B	∅	X	∅	∅	1	5	0.9	2	1
C	∅	∅	X	∅	4	4	0.9	0	2
E	∅	∅	∅	X	*1*	*3*	*1.0*	*1*	*1*
Codified constraints:					21	22	0.75	3	4

(e) Total project demands

TPT=max(4+max(4,1),1)=8

TPC=2+5+4+3=14

TPQ=(0.8+2*0.9+1.0)/(4*0.9+1.0)= =3.6/4.6=0.78

TPS=1.0+1.0+0.9+1.0=3.7

TPR$_1$=max(r_1(t))=3

TPR$_2$=max(r_2(t))=4

Table 5.
The phases of the computation.

three phases. In the first phase, binary decisions select supplementary tasks to include the project plan. The excluded tasks' demands and dependencies (i.e., rows and columns in the PDM) are also ignored. The result is a *project scenario*, which still contains flexible dependencies, but it is decided that all tasks will be completed. Phase two decides flexible dependencies to include or exclude, or in other words, the flexible dependency between tasks either to be specified or to be ignored. A single (multilevel) *project structure* is the result of this phase, which contains only fixed dependencies. In the last phase, we obtain traditional project plans, which must be solved by standard project scheduling algorithms. The result of this phase is the single (multilevel) *project schedule*, where the completion modes and start times of all tasks are specified.

Table 5 shows an example for the computation process of a single project, where the target function is the minimal project duration. Task E is unplanned, which means, the constraints are specified before task E occurs. **Table 5(a)** shows the original project plan, where there are 3 completion modes, 2 renewable resources, cost demands, and quality parameters. There is no nonrenewable resource, however, the cost demand can be considered as a special nonrenewable resource. While keeping constraints, the algorithm has to find a minimal project duration.

The minimal TPT occurs if all flexible tasks and dependencies are excluded, however, neither the scope nor the quality constraint does not allow to exclude all tasks (see **Table 5(b)**), but only task D. In addition, all flexible dependencies cannot be excluded. For example, in the case of the parallel execution of task A and task C, the resource constraint cannot be kept (see **Table 5(c)**). The algorithm excludes the infeasible structures in phases 1–2. The result of phase 2 provides a multimode resource-constrained project scheduling problem, which can be solved by traditional scheduling algorithms. The final result is a project schedule matrix (PSM) (see **Table 5(d)**), where both the structure of the project plan and its demands are specified (see **Table 5(e)**).

4. Risk evaluation

Kosztyán et al. [9] proposed a flexible matrix-based method for risk evaluation for single projects, where all the *risk factors*, such as changes of durations, changes of resource demands, or even the changes of the task priorities; *risk effects*, such as delays, overbudgets, etc.; *stakeholders*, such as managers, vendors and developers and their *goals*, such as minimal project duration, minimal costs, maximal quality, etc., and their inter-dependencies can be modeled. Nevertheless, phases of risk evaluation can be extended to the multiproject level.

The proposed survival analysis-based risk evaluation (SABER) contains three stages (or phases). In all phases, the feasibility of the projects is checked. *A project plan is a surviving project plan if it is still feasible at the end of the risk evaluation process.* The evaluation process covers the preparatory, planning, and tracking stages. With SABER, the agents can be compared and competed with each other. The decision maker decides whether a traditional approach, such as TPMa, or a flexible approach, such as APMa, EPMa, or even the hybrid (HPMa) approach, should be applied. The SABER contains the following stages:

1. Since before starting projects, the boundary conditions of the project are agreed upon, at *stage one*, the effects of changing constraints are examined.

2. At stage two, a two-step Monte Carlo Analysis (MCA) is applied. In the first step, the set of tasks and the relationship of risk factors are selected, while in the second step, the changes in demands and priorities are changed for the selected tasks.

3. At *stage three*, the two-step MCA is also applied, but only for the remaining and running tasks of running projects.

Figure 1 shows the simulation framework of the SABER.

The simulation framework indicates, which project plans survive. It shows the performances of applied agents and sensitivities of risk effects, and in addition, the interdependency of risk factors and risk effects, see in detail in [9].

5. Computer applications

There are free available matrix-based project planning tools for flexible projects [10]. This plug-in can be applied both for project planning and for risk evaluation. It contains 5 domains, such as LD, TD, CD, QD, and RD. It solves several project scheduling problems (PSPs) for flexible projects, such as Pareto-optimal (multimode) resource-constrained PSPs for a single target function and their Pareto-optimal solutions for multiple targets.

6. Application examples

The application example guides us through matrix-based planning and risk evaluation phases.

The first step is to specify a matrix-based project plan. It can come from the original network-based project plan, but it can be generated. In this case, at least the number of tasks, number of modes, and number of resources have to be specified.

Figure 2(a) shows the logic structure of the flexible project plan. The minimal structures and their demands contain only mandatory tasks and fixed dependencies, while the maximal structure and its demands save all tasks and dependencies, and therefore, all demands. **Figure 2** shows the minimal/maximal demands.

Figure 2 shows that there are significant differences between the minimal and maximal structures and their demands. Flexible and hybrid approaches can

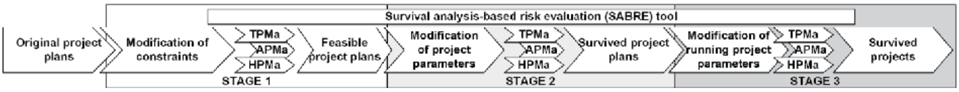
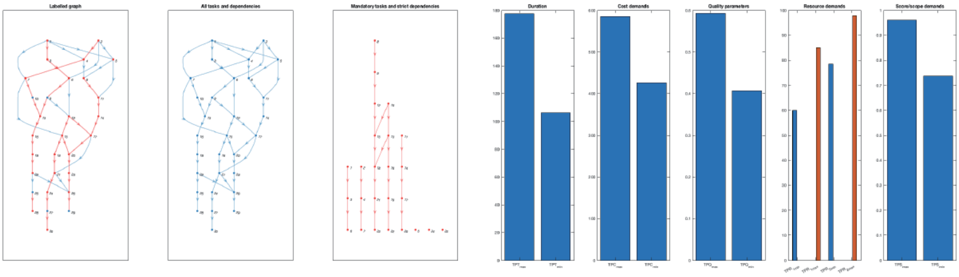

Survival analysis-based risk evaluation (SABRE) tool
Original project plans
Modification of constraints
TPMa
APMa
HPMa
Feasible project plans
STAGE 1
Modification of project parameters
TPMa
APMa
HPMa
Survived project plans
STAGE 2
Modification of running project parameters
TPMa
APMa
HPMa
Survived projects
STAGE 3

Figure 1.
The simulation framework.



(a) Logic structures

(b) Minimal/maximal project demands

Figure 2.
Minimal/maximal structures and demands (the number of tasks was 30, and the number of completion modes and the number of renewable resources was 2).

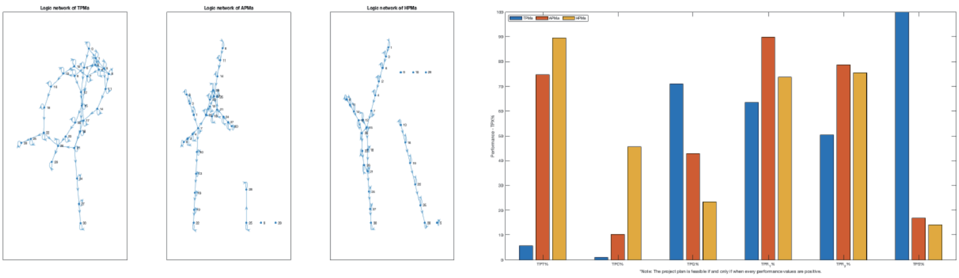
reorganize project structures and reprioritize task completion to save the project plan as feasible. **Figure 3(a)** shows the result project structures of the project management approaches. **Figure 3(b)** compares the project demands. The target function was the minimal distance from the minimal demands, while the constraint was 2/3 of the maximal demands. The vertical axis of 3(b) is the performance ($TPX\%$) of the project plans, which is the relative distance between the minimal demands.

$$TPX\% = \frac{TPX - TPX_{\min}}{TPX_{\max} - TPX_{\min}} \tag{20}$$

If $TPX\% \in [0, 1]$ for all demands, then the project plan is feasible.

Figure 3 shows that the flexible and hybrid project management approaches try to parallelize task completions and exclude low priority tasks, while TPMa tries to reduce demands but keeps all tasks.

Figure 4 shows the comparison of scheduling performances of the project management approaches under risks. **Figure 4** shows that in this case, only the HPMa ensures the survival of the project plan.



(a) Logic networks

(b) Project demands

Figure 3.
The comparison of project management approaches.

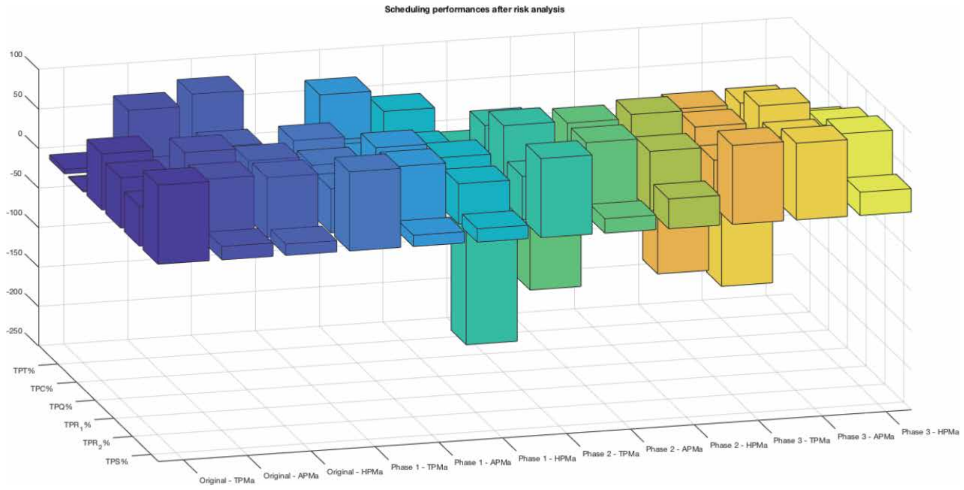


Figure 4.
The scheduling performances of project management approaches under risks.

7. Conclusions

In this chapter, matrix-based approaches are proposed to schedule traditional, flexible, and hybrid project plans. In addition, a risk evaluation tool is proposed to compare project management approaches. It is important to note that there is usually no superior project management approach. If the goal is to complete all tasks, traditional approaches are required; however, flexible projects require flexible project management approaches. Nevertheless, hybrid approaches can better ensure the survival of the project. The study also offers freely available matrix-based project planning applications; therefore, all traditional, flexible, and hybrid project management, planning and scheduling approaches can be supported.

Conflict of interest

The authors declare no conflict of interest.

Nomenclature

CD	cost domain
LD	logic domain
ND	nonrenewable resource domain
QD	quality domain
RD	renewable resource domain
TD	time domain

Variables

n	number of tasks
m	number of completion modes
p	number of projects
η	number of nonrenewable resources
ρ	number of renewable resources
τ	actual time of a running project

Constraints

C_t	time constraint (upper bound, scalar)
C_c	cost constraint (upper bound, scalar)
C_q	quality constraint (lower bound, scalar)
$\mathbf{C}_n$	nonrenewable resource constraint (upper bound, 1 by η vector)
$\mathbf{C}_r$	renewable resource constraint (upper bound, 1 by ρ vector)
C_s	score/scope constraint (lower bound, scalar)

Simple target functions

$TPT \rightarrow \min$	minimize project duration
$TPC \rightarrow \min$	minimize project cost

$TPQ \to \max$	maximize project quality
$TPN_e \to \min$	minimize the sum of nonrenewable resources e, $e = 1, .., \eta$
$TPR_r \to \min$	minimize the maximum of renewable resources r, $r = 1, .., \rho$
$TPS \to \max$	maximize project score/scope

Abbreviations

APM(a)	agile project management (agent)
DMM	domain mapping matrix
DSM	design/dependency structure matrix
EF	early/earliest finish time
EPM(a)	extreme project management (agent)
ES	early/earliest start time
HPM(a)	hybrid project management (agent)
LF	late/latest finish time
LS	late/latest start time
M^5	matrix-based multimode multilevel (project) management model
MCA	Monte Carlo analysis [MDM] multidomain matrix
MPE	reverse extreme project management
MPR	multilevel project ranking algorithm
NDSM	numerical dependency structure matrix
PDM	project domain matrix
PSP	project scheduling problem
PSM	project schedule matrix
SABER	survival analysis-based risk evaluation
SF	scheduled finish time
SS	scheduled start time
TPC	total project cost
TPM(a)	traditional project management (agent)
TPN	total project nonrenewable resources
TPQ	total project quality
TPR	total project renewable resources
TPS	total project scenario score
TPT	total project time

Author details

Zsolt T. Kosztyán
University of Pannonia, Veszprém, Hungary

*Address all correspondence to: kosztyan.zsolt@gtk.uni-pannon.hu

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

References

[1] Wysocki RK. Effective Project Management: Traditional, Agile, Extreme. Indianapolis, Indiana, USA: John Wiley & Sons; 2019

[2] Kosztyán ZT. Exact algorithm for matrix-based project planning problems. Expert Systems with Applications. 2015;**42**(9):4460-4473. DOI: 10.1016/j.eswa.2015.01.066

[3] Servranckx T, Coelho J, Vanhoucke M. Various extensions in resource-constrained project scheduling with alternative subgraphs. International Journal of Production Research. 2021:1-20. DOI: 10.1080/00207543.2021.1924411

[4] Danilovic M, Browning TR. Managing complex product development projects with design structure matrices and domain mapping matrices. International Journal of PROJECT MANAGement. 2007;**25**(3): 300-314. DOI: 10.1016/j.ijproman.2006.11.003

[5] Browning TR. Design structure matrix extensions and innovations: A survey and new opportunities. IEEE Transactions on Engineering Management. 2015;**63**(1):27-52. DOI: 10.1109/TEM.2015.2491283

[6] Kosztyán ZT, Szalkai I. Hybrid time-quality-cost trade-off problems. Operations Research Perspectives. 2018; **5**:306-318. DOI: 10.1016/j.orp.2018.09.003

[7] Kosztyán ZT, Szalkai I. Multimode resource-constrained project scheduling in flexible projects. Journal of Global Optimization. 2020;**76**(1):211-241. DOI: 10.1007/s10898-019-00832-8

[8] Kosztyán ZT. An exact algorithm for the flexible multilevel project scheduling problem. Expert Systems With Applications. 2020;**158**:113485. DOI: 10.1016/j.eswa.2020.113485

[9] Kosztyán ZT, Jakab R, Novák G, Hegedüs C. Survive IT! Survival analysis of IT project planning approaches. Operations Research Perspectives. 2020;**7**: 100170. DOI: 10.1016/j.orp.2020.100170

[10] Kosztyán ZT. MFPP: Matrix-based flexible project planning. SoftwareX. 2022;**17**:100973. DOI: 10.1016/j.softx.2022.100973

[11] Creemers S. Minimizing the expected makespan of a project with stochastic activity durations under resource constraints. Journal of Scheduling. 2015;**18**(3):263-273. DOI: 10.1007/s10951-015-0421-5

[12] Chun-Hsien C, Fu LS, Wei C. Project scheduling for collaborative product development using DSM. International Journal of Project Management. 2003;**21**(4):291-299. DOI: 10.1016/S0263-7863(02)00023-6

[13] Steward DV. The design structure system: A method for managing the design of complex systems. IEEE Transactions on Engineering Management. 1981;**3**:71-74. DOI: 10.1109/TEM.1981.6448589

[14] Kuhrmann M, Diebold P, Münch J, Tell P, Trektere K, Caffery FM, et al. Hybrid software development approaches in practice: A European perspective. IEEE Software. 2018;**36**(4): 20-31. DOI: 10.1109/MS.2018.110161245

[15] Tyagi MK, Munisamy S, Reddy LSS. Traditional and hybrid software project tracking technique formulation: State space approach with initial state uncertainty. CSI Transactions on ICT. 2014;**2**(2):141-151. DOI: 10.1007/s40012-014-0037-5

[16] Rahimian V, Ramsin R. Designing an agile methodology for mobile software development: A hybrid method engineering approach. 2008 Second International Conference on Research Challenges in Information Science. June 3-6, 2008, Marrakesh, Monaco: IEEE; 2008. pp. 337-342. DOI: 10.1109/RCIS.2008.4632123

Chapter 4

Particle Swarm Optimization of Convolutional Neural Networks for Human Activity Prediction

Preethi Gunishetty Devarakonda and Bojan Bozic

Abstract

The increased usage of smartphones for daily activities has created a huge demand and opportunities in the field of ubiquitous computing to provide personalized services and support to the user. In this aspect, Sensor-Based Human Activity Recognition (HAR) has seen an immense growth in the last decade playing a major role in the field of pervasive computing by detecting the activity performed by the user. Thus, accurate prediction of user activity can be valuable input to several applications like health monitoring systems, wellness and fit tracking, emergency communication systems etc., Thus, the current research performs Human Activity Recognition using a Particle Swarm Optimization (PSO) based Convolutional Neural Network which converges faster and searches the best CNN architecture. Using PSO for the training process, intends to optimize the results of the solution vectors on CNN which in turn improve the classification accuracy to reach the quality performance compared to the state-of-the-art designs. The study investigates the performances of PSO-CNN algorithm and compared with that of classical machine leaning algorithms and deep learning algorithms. The experiment results showed that the PSO-CNN algorithm was able to achieve the performance almost equal to the state-of-the-art designs with a accuracy of 93.64%. Among machine learning algorithms, Support Vector machine found to be best classifier with accuracy of 95.05% and a Deep CNN model achieved 92.64% accuracy score.

Keywords: Human Activity Recognition, Particle Swarm Optimisation, Convolutional Neural Network, Time Series Classification, Deep Learning, Sensors

1. Introduction

1.1 Background

Activity Recognition aims at identifying the activity of users based on series of observations collected during the activity in a definite context environment. Applications that are enabled with activity recognition are gaining huge attention, as users get personalized services and support based on their contextual behavior. The proliferation of wearable devices and smartphones has provided real-time monitoring of human activities through sensors that are embedded in smart devices such as proximity sensors, cameras, microphone, magnetometers accelerometers, gyroscopes, GPS etc., Thus, understanding human activities in inferring the gesture or position has created a competitive challenge in building personal health care systems, examining wellness and fit characteristics, and most pre-dominantly in elderly care, abnormal activity detection, diabetes or epilepsy disorders etc.,

Thus, Human Activity Recognition (HAR) plays a significant part in enhancing people's lifestyle, as it should be competent enough in learning high level quality information from raw sensor data. Effective HAR applications are incorporated for contextual behavior analysis [1], video surveillance analysis [1], gait investigation (to determine any abnormalities in walking or running), gesture and position recognition.

1.2 Research problem

Human Activity Recognition (HAR) is evolving to be a challenging time series classification task which involves predicting the human activity based on sensor data where the data points are recorded at regular intervals. Though HAR seems to be the straightforward approach of performing HAR, there are numerous issues and challenges that are encountered in selecting the appropriate feature processing technique and thus choosing the correct modeling algorithm for the time series data is crucial. Thus, the conventional approaches have made extraordinary progress on Human Activity Recognition (HAR) by incorporating machine learning algorithms such as Naïve Bayes, Decision Tree, Support Vector Machine, Logistic Regression as there are only few labeled data. It requires domain knowledge to manually process the feature extraction. On the other hand, deep learning algorithms has seen high performance in areas like Natural language Processing, Object Recognition etc., In spite of these advancements, another line of research has emerged in applying nature-inspired meta heuristic optimization techniques like Particle Swarm Optimization, Genetic Algorithms on Neural Networks. The research question that is aimed to be addressed in the current study can be concisely stated as below.

"To what extent can the Particle Swarm Optimized Convolutional Neural Network significantly enhance the recognition of human activity from raw inertial sensor data when compared with supervised machine learning algorithms and Deep Learning Algorithms".

Algorithms: Naive Bayes, Support Vector Machine (SVM), Random Forest (RF) Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), PSO Optimized Convolutional Neural Network.

1.3 Research objectives

The most feasible solution in overcoming these challenges could be by looking into existing works and analyzing the experimental set up. Thus, picking the right sensor and right gestures with demonstrated capabilities can significantly eliminate the chances of inaccurate sensor data. The traditional machine learning algorithms require large amount of labeled static data and manually performing the feature selection tasks. But in real applications most of the activity data are unlabelled and entire data needs to be analyzed. Since Deep Learning methods can perform training on the entire data, and analyzing the complex features, this study focuses on investigating the performance various deep learning models in classifying the time series data.

Convolutional Neural Network requires large number of parameters to tune and it is time consuming. The study explores the optimization using metaheuristic algorithm Particle Swarm Optimization for Convolutional Neural Network. Thus, the study has a deep investigation towards major approaches followed in HAR namely, machine learning using hand-crafted features and deep learning using raw inertial signals. The process in which the research carried to achieve the results is mentioned below.

1. Exploring the previous works on Human Activity Recognition, identifying gaps in the research with a detailed analysis.

2. Data Preparation and Data Pre-Processing is conducted by considering two versions of the dataset.

3. Designing a solution to perform Human Activity Recognition by using Particle Swarm Optimized Convolutional Neural Network.

4. Implementing the solution fortified in the proposed design and tune the models to obtain the expected accuracy.

5. Evaluate the performance of the various models.

6. Comparing the results obtained for different models and place the findings in the study.

2. Review of existing literature

This section provides an overview on Human Activity Recognition and its applications. Various approaches for HAR task are discussed. Particularly, Sensor based HAR is detailed with different sensor modalities. Additioanlly gives an overview of the modeling approaches for HAR.

Sensor Based Human Activity Recognition: Due to the immense growth of sensor technology and ubiquitous computing, sensor-based Human Activity Recognition is gaining attention which is widely used with enhanced protection and privacy. According to [2], the HAR task can be achieved by placing the sensors at different locations to recognize human activity for specific context **Table 1** based on sensor placements at different locations. HAR with different sensor modalities are listed below.

Body-worn sensors/wearable sensors Wearable sensors are one of the widely used sensor modalities in HAR. These sensors are often worn or attached to the users, namely an accelerometer, gyroscope, and magnetometer. As the human body moves, the acceleration and angular velocity are varied, this data is further analyzed to predict the activity. These sensors can be embedded in smart phones, smart watches, fit bands, headbands etc., The **Figure 1** shows the different wearable sensors that can be used by humans [4, 5] studies the significance of sensor and its appropriate position to be placed on the body of the user. Many research are conducted to investigate the variability in accuracies by placing the sensors on

Modality	Description	Sensor Types
Wearable	Usually Worn by the user to capture the body movements	Smartphone, watches gyroscope, accelerometer
Object Sensors	Mounted on objects to capture objects movements	RFID, accelerometer on objects
Ambient Sensors	Mounted in environment surroundings to record user interaction	Bluetooth,Sound,WiFi,
Hybrid Sensors	Combination of multiple sensors	Multiple types, often deployed in smart environments

Table 1.
Sensor modalities for human activity recognition tasks [3].

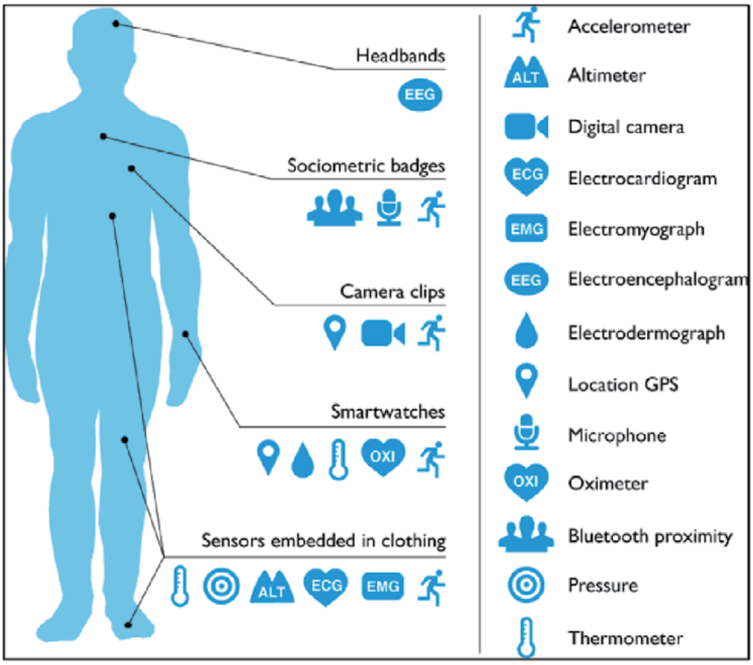


Figure 1.
Body worn sensors [4].

different pasts of human body. One such study is performed by [6] by placing sensors on chest and wrist for duration of two hours which gave 83% classification accuracy.

Thus, wearable sensors were widely used for HAR [7] in various health monitoring systems. In recent days, inertial sensing, that uses movement-based Sensors which can be attached on user's body has been studied widely [8]. Among those work, the accelerometer is mostly used sensor for collecting position details. Gyroscope and magnetometer are also used in combination with accelerometer.

2.1 Modeling approaches for human activity recognition

In any data mining project, the choice of the appropriate modeling algorithm does not depends only on the type of problem to solve, but also on the type of input data. Due to the natural ordering of the temporal feature data, the Human Activity Recognition is considered as a typical pattern-recognition system where it involves classifying the human activity based on the series of data. The main difference between Machine Learning Algorithms and Deep Learning Algorithms in recognizing human activity is the way the input features are extracted. In this aspect, the below sections explain the methodology chosen for the task of Human Activity Recognition.

Machine Learning Algorithms for Human Activity Recognition: Considering HAR as one of the pattern recognition problem, the conventional pattern recognition methods have seen extraordinary results by utilizing machine learning algorithms like hidden Markov models, decision tree support vector machine, naive Bayes [9]. The **Figure 2** illustrates the process of Human Activity Recognition using hand-crafted features modeled with machine learning algorithms. The raw inertial activity signals received from the sensors are subjected to feature -extraction process by domain knowledge experts [10]. The features that are usually extracted are based on two main domain features namely; time domain and frequency domain [11]. The time

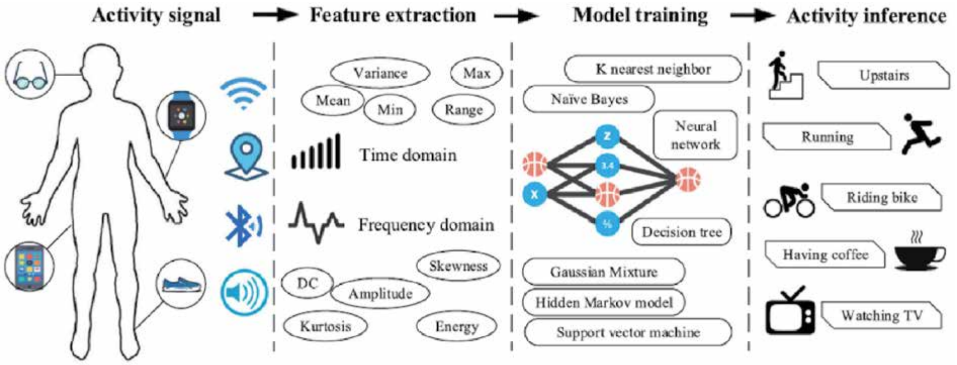


Figure 2.
Process of human activity recognition using hand-crafted features modeled with machine learning algorithms.

domain features are computed based on mathematical functions to extract statistical details from the signals. The frequency domain features possess mathematical functions that record recursive patterns of signals. Thus, in machine leaning approach for HAR, the input data are always extracted from human engineered hand-crafted features. These features may be further pre-proceed using Data Dimensionality Reduction techniques to select the significant features.

Selecting important features is more significant than choosing a classification algorithm [12], this is because poor quality features may hinder the performance of the classifier. Hassan et al. [13] in his recent work, employed Kernel Principal Component Analysis (KPCA) which works based on statistical analysis before applying modeling. Furthermore, [14] employed Stepwise Linear Discriminant Analysis (SWLDA) which is a non-linear method, selects the subset of features by using regression combined with F-test. The model showed enhanced performance after applying Data Dimensionality Technique.

Different modeling algorithms have been employed to predict the human activity recognition. Ravi et al. [15] in his work used Naïve Bayes classifier with few parameter settings to classify 8 different activities, which outperformed other classification algorithms. Several research employed Naive Byes as the primary classifier for human activity recognition [16, 17].

In recent times, learning algorithms which are based on error computation namely; Artificial Neural Networks [18, 19] Support Vector Machine [20, 21] new are used for predicting HAR without any Data pre-processing technique applied.

The most used modeling algorithms that showed efficient results as per the study are Naive Bayes, Multinormal Logistic Regression, K - Nearest Neighbor Hidden Markov Models, Support Vector Machine and Artificial Neural Network.

Deep Learning Approaches for Human Activity Recognition: Though, conventional Pattern Recognitions (PR) approaches gained satisfactory results in HAR, these methods heavily rely on hand crafted feature generation usually done by domain expertise [22]. This sometimes leads to error in collection data and missing some significant data points. On the other hand, Deep Neural Networks are capable of automatic feature extraction without human intervention. In fact, the model becomes more robust when data is large [23].

The **Figure 3** illustrates the process of HAR followed by Deep Learning Algorithms. Initially, the raw sensor signals collected from inertial sensors (accelerometer, gyroscope etc.,) are it is directly subjecting to modeling, where no feature extraction step is performed. Additionally, deep learning follows a unsupervised, incremental learning which makes it more feasible to implement HAR tasks [24].

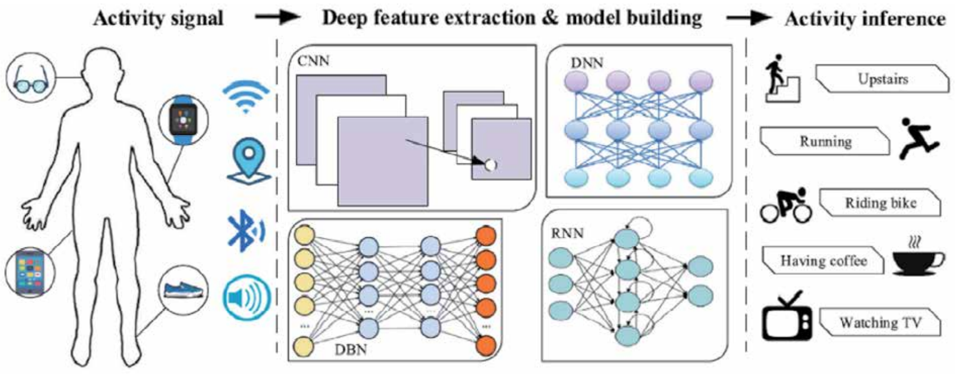


Figure 3.
Process of human activity recognition using raw inertial signals modeled with deep learning algorithms.

Several Deep learning mode were employed to perform Activity Recognition in various contexts. Liu et al. [25] investigated the performance of Restricted Boltzmann Machines for Activity Recognition from data collected through smart watches. The method outperformed other models and gained high accuracy results with less computation time.

Additionally, Long short-term memory (LSTM) models has been utilized to predict the activity performed for unbalanced real world data where the model performance wss evaluated using f1 score due to imbalance nature of data [26]. Vepakomma et al. [27] the hand-crafted features are obtained from inertial sensors, and these features are added into DNN algorithm. In this aspect, [28] used PCA as a Dimensionality Reduction Technique before modeling to Deep Neural Network (DNN). However, since domain knowledge is used for feature extraction, the model cannot be generalized.

Some works used Recurrent Neural Network (RNN) for the HAR [26, 29], where the learning rate and computational power are the main constraints. More time is invested in finding the optimal set of hyper parameters that provides the best results. Inoue et al. [30] identified various model parameters and recommended a model that would achieve high accuracy of HAR by turning the hyperparameters. The main constraint of RNN based Human Activity Recognition models is to deal with the time, power constraint environment, while still thriving to achieve good performance results.

Furthermore, CNN's are used more extensively for HAR tasks with varied experimental settings. In general, CNNs are mostly used for image classification using 2-Dimensional Convolution since it accepts the data with shape n * n. Several works resized the single dimension input data to a 2D image so as to make use of 2D convolution. Ha et al. [31] used similar approach in reshaping the input data to a 2D image. While [7] designed a complex design of CNN algorithm by transforming time series data into an image. Other works include [32, 33] performed data transformation to achieve CNN model driven results.

3. Why particle swarm optimization?

As seen from the above section, deep learning networks have gained better results with less efforts in parameter settings. In particular, Deep Convolutional Neural Networks are used extensively due to its flexibility in both data driven approach (Using 1D Convolution for signal data) and model driven approach (data transformation of signal data to a 2D image). In order to gain higher performance of the model, several layers has to be used and parameter initialization has to be done carefully. This needs a detailed knowledge on CNN architecture and also on the dataset.

Thus, to find the optimal CNN architecture automatically without human intervention, a meta heuristic algorithm Particle Swarm Optimisation is utilized which is easy to implement with lower computational cost.

Theory:

Particle Swarm Optimization (PSO) is a nature inspired, meta-heuristic algorithm often used for discrete, continuous and sometimes for combination optimization problems. The PSO was first introduced by Kennedy and Eberhart in 2001 [34] which is inspired by the pattern followed by a flock of words during flying. PSO works by making only few or no assumptions regarding the problem being optimized and possess the ability to search large spaces of candidate solutions in an efficient manner.

In PSO, a particle is called a single solution and the total of all such solutions is termed as swarm. The main ideology behind PSO is that each particle is well known of its velocity and the best configuration achieved in the past (pBest), and the particle which is the current global best configuration in the swarm of particles (gBest). Hence, at every current iteration, each particle updates its velocity in such a way that its new position will be close enough to global gBest and its own pBest at the same time. The velocity and particle vector are adjusted based to the following Eqs. 2.1 and 2.2 respectively:

$$v_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (P_{id} - x_{id}(t)) + c_2 * r_2 * \left(P_{gd} - x_{id}(t)\right) \qquad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \qquad (2)$$

where v_{id} indicates the velocity of ith particle in the dth dimension, z_{id} indicates the position of i_{th} particle in the d_{th} dimension, P_{id} and P_{gd} represents the local best and the global best in the dth dimension, r_1 and r_2 are the random numbers between the range 0 and 1, c_1, c_2 and w, are acceleration coefficient for exploitation,acceleration coefficient for exploration and inertia weight respectively. Since the encoded vector in the proposed method is fixed-length and consists of decimal values, and PSO is effective to search for the optimal solution in a fixed-length search space of decimal values, the proposed method will use PSO as the search algorithm. One of the advantages of PSO is that they converge at a faster rate than Genetic Algorithms (GA).

Gaps in the Research:

The literature review outlines the existing works on Human Activity Recognition in terms of the modeling approaches chosen. However, certain gaps are found in both the approaches.

Some research works employed Machine Learning Approach to perform HAR with hand-crafted features faced low performance as only shallow features are explored and learned by the classifiers. Before deep learning was used extensively, shallow neural network classifiers, that is Multi-Layer Perceptron (MLP), was considered to be a promising algorithm for HAR. In this aspect, [35] performed HAR with algorithms like logistic regression, decision tree and MLP and MLP outperformed the other two models.

As deep convolutional neural networks (CNNs) has been used to obtain the excellent results in most of the image classification benchmarks datasets, they have overcome the need of human experts for classification. But still, it remains a challenging task to find the meaningful CNN architecture that would apply for all type of domains. As a result of some of the successful CNN's architecture like ResNet [36], VGG16 [37], DenseNet [38] were introduced recently considering domain knowledge. The results from this outperformed the state-of-the-art baseline CNN model. However, the CNN's architecture are designed by doing lot of trial and error methods and are suitable to handle problems only in specific context.

PSO algorithm was employed to train an Artificial Neural Network by [39]. The results show that ANN's training time was reduced with PSO greatly. In this aspect, [40] designed PSO algorithms for two tasks that is to train ANN and to find better architecture. This resulted in achieving competitive result than other models.

Thus, most of the works in PSO was used to find optimal architectures in full connected networks [41], but these cannot be used for tasks like image classification, activity recognition which indeed used a complex deep layers. Human Activity Recognition. Recently, [42] came up with PSO trained for CNN architecture that is suitable for only image classification with 2-Dimensional Convolutions. The experiment was performed with 10 benchmark datasets, and the results were outstanding.

However, there is not much work done on using 1-Dimensional Convolution for finding optimal architectures in CNN. Considering the gaps in the literature review, the current research aims to address few issues and find solutions that generalizes the models for Human Activity Recognition tasks.

Research Question: Thus, considering the gaps in the above mentioned literature review, this research aims to address the below research question.

"To what extent can the Particle Swarm Optimized Convolutional Neural Network significantly enhance the recognition of human activity from raw inertial sensor data when compared with supervised machine learning algorithms and Deep Learning Algorithms".

4. Experiment design and methodology

This chapter gives the plan and the research methodology used for performing the research. The Cross Industry Standard Process for Data Mining (CRISP–DM), a well proven methodology with a structured approach (Piatetsky, 2014) is employed to conduct the current study.

The **Figure 4** shows the design flow to be followed for the current research. The experiment begins with the Business Understanding phase, which indicates what is to be accomplished from a business perspective. The expected outputs of this phase

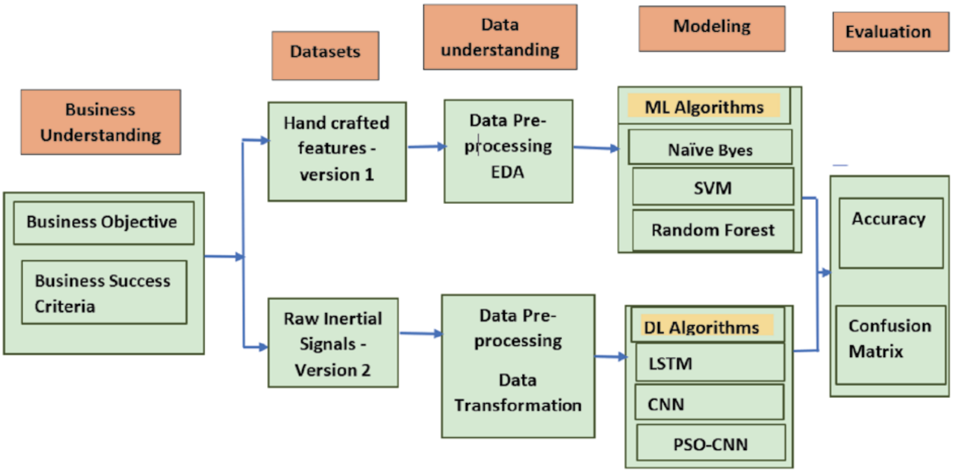


Figure 4.
Design flow.

form the main objectives of the project. Here the insights and goals of the project are defined. In order to answer the research question, the experiment is conducted with two versions of the datasets which is explained in Data Understanding Phase. Additionally, data description report is prepared to understand each filed description. This is done separately for both the datasets.

The third phase is the Data preparation stage. Here the data is checked for duplicates, null records and appropriate action is taken to address them. Further, new derived fields can be formed based on the domain knowledge. Data from multiple databases are integrated to form the final dataset for modeling. The fourth step is the modeling stage.

Based on initial analysis done from the literature review, suitable modeling technique is chosen and applied on the two versions of the dataset. Next phase is Evaluation phase. Based on the evaluation criteria, models are evaluated to see if it meets the business objective.

Business Understanding: The primary aim of the project is to identify a best classification algorithm which identifies the different human activities in motion accurately. Thus, the predicted activity can be applied to multiple applications like health monitoring and controlling systems, wellness and fit tracking, alarming to emergency situations etc.,

- Business Success Criteria: The solution for the research problem must not only find the models which performs in classifying target data, but also ensure to show the confirmation that the results obtained are significant and is consistent when tried to repeat the solution. By considering the above business objectives, evaluation criteria and constraints, below hypothesis is formed to answer the research question.

Hypothesis 1

- *Null Hypothesis (H0):* If Particle Swarm Optimized Convolutional Neural Network is used instead of supervised machine learning then there is no significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

- *Alternate Hypothesis (HA):* If Particle Swarm Optimized Convolutional Neural Network is used instead of supervised machine learning, then there is significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

Hypothesis 2

- *Null Hypothesis(H0):* If Particle Swarm Optimized Convolutional Neural Network is used instead of deep learning algorithms, then there is no significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

- *Alternate Hypothesis (HA):* If Particle Swarm Optimized Convolutional Neural Network is used instead of deep learning algorithms, then there is significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

Data Understanding: The dataset used in this study is downloaded from UCI Machine Learning Repository created at SmartLab, one of the Research Laboratories at DIBRIS at University of Genova. (Anguita, 2006) experimented on a group of 30 volunteers within a range of age between 19 and 48 years who were performing daily activities. Each subject are volunteer performed daily activities which are monitored while carrying a smartphone (Samsung Galaxy S II) that is waist-mounted. The smartphone was embedded with inertial sensors.

With the help of this embedded accelerometer and gyroscope, 3-axial linear acceleration and 3-axial angular velocity were captured at a constant rate of 50 Hz. To label the data manually, the experiments are captured in the form of video. Additionally, the sensor signals accelerometer and gyroscope were processed by including noise filters and sampled with fixed width sliding windows of 2.56 seconds and 50% overlap that indicates 128 readings per sliding window. The resulting processed signals is a combination of gravity acceleration and body acceleration components and were subjected into a low pass filter to get the separated components with the gravitational force components cut off at the lower end of the filter. As such, vectors are formed from each window to obtain time and frequency domain variables.

Thus, each record in the dataset includes below features:

- total acceleration and approximate body acceleration which is obtained from Triaxial Accelerometer.
- Triaxial Angular velocity is extracted from gyroscope.
- A 561-feature vector with time and frequency domain variables.
- activity as target.
- An identifier which indicates the participant who performed the experiment.

Data Gathering: Two versions of the data was made available for modeling purposes. These are mentioned as follows

- Hand-crafted features of activity windows- Version 1: Each recorded window possess a 561 column vector with time and frequency domain variables rectified separately, a activity label ID indicating the activity performed by the subject and an identifier or the ID of the subject who carried out the experiment.
- Raw Inertial sensor data - Version 2: Raw signals which are tri-axial from the gyroscope and accelerometer sensor are collected by placing wearable sensors on the volunteer called subjects.

4.1 Modeling

The **Figure 5** shows the distribution of target activities. Modeling algorithms are sued to predict the target class.

4.1.1 Machine learning algorithms – with hand-crafted features – Version 1

As discussed, Version −1 dataset is modeled with classical machine learning algorithms. Each modeling algorithms along with the parameter settings are discussed below.

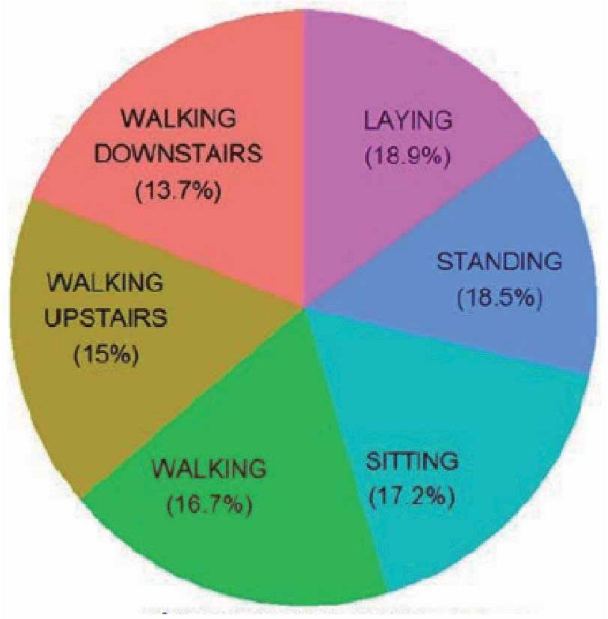


Figure 5.
Distribution of target class.

Naïve Byes: Naïve Byes classifier implements Bayes Theorem providing probabilistic classification [30]. This is suitable for fast computation especially in huge data.

Random Forest: Random Forest Classifier is nothing but a combination of multiple decision Trees which is an ensemble learning method for classification, regression and other machine learning tasks. This performs training by building multiple decision trees and designating the output of the class which is mode in case of classification of the individual trees.

Support Vector Machine: Support Vector Machine is one of the baseline models which gained highest accuracy in human activity recognition when compared with other classical machine learning algorithms. SVM is implemented for both classification and regression tasks and this works by building builds the hyperplane margin between classes.

4.1.2 Deep learning algorithms – With raw inertial signals – Version 2

Sensor based activity recognition requires domain-level knowledge about human activities to analyze even the minute details of sensor data. Though traditional machine algorithms have shown some extra-ordinary performance in classifying human activities, it requires domain knowledge and few labeled data. In contrast, Deep Learning exhibits the capability of training real time activity data that are coming in stream or sequence. Considering Human Activity Recognition as a Time Series Classifications problem which aims at classifying sequences of sensor data, two well-known algorithms LSTM and CNN are modeled on the raw inertial signal data - Version 2.

Long Short-Term Memory (LSTM): Long Short-Term Memory networks – are a special kind of Recurrent Neural Network (RNN), capable of learning long-term dependencies. LSTMs are designed to avoid the long-term dependency problem. The LSTM learns to map each sliding window of sensor data to an activity, where the data points or samples in the input sequence are read one at a time, and each time step may consist of one or more variables.

Convolutional Neural Network (CNN): CNN has achieved good results in image classification, sentiment analysis and speech recognition task by extracting features from signals. CNN has been used for time series classification problems especially in classifying real time activity data because of scaling invariable and local dependencies. Local dependency means the nearby signals in Human Activity Recognition (HAR) are likely to be correlated to each other, while scale invariant means that the scale remains same for different time and domain frequencies. Thus, CNN has a better understanding of learning features that are present in recursive patterns.

Particle Swarm Optimization Based CNN: Though CNN's have showed good results in HAR, there are multiple parameters to take care to find the optimal CNN architecture. The main focus of any neural network is to minimize the error between training targets and predicted outputs. It is cross-entropy in case of CNN's, which is carried out by backpropagation and gradient descent. Even a simple CNN's have many parameters to tune them. Thus, it is significant to find algorithms which finds and evaluates CNN architecture with less time. Thus, motivated from this, a new PSO-CNN is utilized for Human Activity Recognition. The below **Figure 6** shows the working of the model.

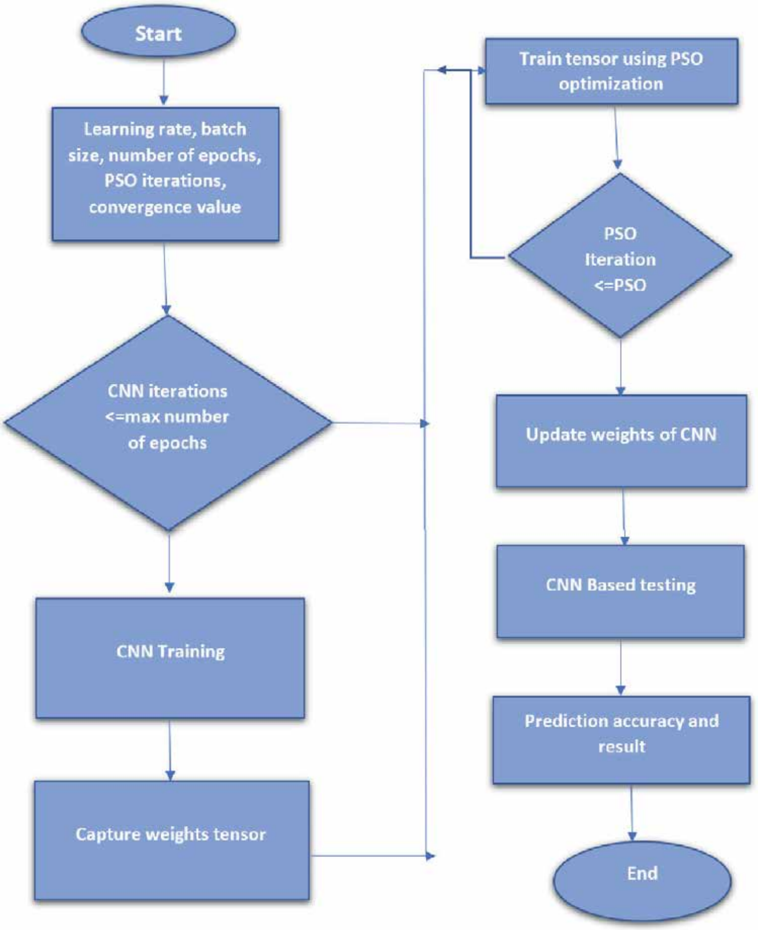

Start
Learning rate, batch size, number of epochs, PSO iterations, convergence value
CNN iterations <=max number of epochs
CNN Training
Capture weights tensor
Train tensor using PSO optimization
PSO Iteration <=PSO
Update weights of CNN
CNN Based testing
Prediction accuracy and result
End

Figure 6.
Particle swarm optimization training for convolutional neural network.

The working of PSO-CNN can be divided into five stages as below

- CNN Training – The CNN is trained with some pre-denied weights initialixed.. It uses a CNN with 1D convolutional layer, since the HAR dataset consist of signals in shape [samples, time_steps, no of features]. The output is one hot vector encoded which is 6 (target activity to be predicted).

- Pre-PSO Training – Here weights are captured from CNN training and it is converted to particle.

- Particle Swarm Optimization Training - After initializing the values of convergence, cognitive value, social value, number of particles, stopping condition and number of epochs, PSO algorithms searches the hyperplane for optimized vector using the CNN loss function .

- Update CNN Architecture - Using the values of weight in previous phase, the final results are computed. A new CNN architecture is created is created based on these weights rather than basis of the output.

- Computation of Prediction Accuracy and Results - The output of the CNN is formed and the final accuracy, loss valued are evaluated.

5. Implementation

Parameter Settings for Particle Swarm Optimization: The parameters used in this category control the behavior of the Particle Swarm Optimisation algorithm. It consists three parameters namely, the number of iterations, the size of the swarm, (Cg) represents the probability of selecting a layer from global best while computing each particle's velocity. The number of iterations specify the actual number of iterations that the optimal search algorithm will run before optimization is completed. The best CNN architecture that is with best accuracy is saved at after the optimization of the last particle. The swarm size indicates number of particles in the PSO algorithm. Here, each individual particle is a one complete CNN architecture whose performance to be tested by the algorithm **Table 2**.

Parameter Settings for initializing CNN architecture: The parameter settings used in the second category control the initial movement of the particles. It involves eight parameters listed in the table below **Table 3**. In this step, an initial population of swarm which is of CNN architectures. This initial population consists of individuals with CNN architectures picked randomly as defined by these parameters. To limit the number of feature maps from a output of a convolution layer, minimum and maximum number of outputs must be defined. The size of the kernel is always chosen will between the range of the minimum and maximum size of a

Description	Value
Number of iterations	10
Swarm Size	20
Cg	0.5

Table 2.
Parameter initialization for particle swarm optimization.

Description	Value
Minimum number of outputs from a conv layer	3
Maximum number of outputs from a conv layer	256
Minimum number of neurons in a FC layer	1
Minimum number of layers	3
Maximum number of layers	20
Minimum size of a Conv kernel	3
Maximum size of a Conv kernel	7

Table 3.
Parameter settings for initializing CNN architecture.

Description	Value
Epochs for particle evaluation	100
Epochs for global best	256
Dropout rate	0.5
Batch normalizer layer outputs	yes

Table 4.
Parameter settings for training convolutional neural network.

convolutional kernel. Only the initial architecture is controlled by these parameters, after first initialization the architecture is updated based on design specified.

Parameter Settings for training Convolutional Neural Network: The parameters here specify the training process of each particle. It includes four parameters to set which is listed in **Table 4**. These parameters control the weight updating process during the training of each particle. The number of epochs specifies the total number of times the particle is trained using entire dataset before its accuracy is evaluated. The dropout parameter is used in the particle to avoid overfitting. Furthermore, the model includes batch normalization between the layers to avoid overfitting during training process.

6. Evaluation metrics

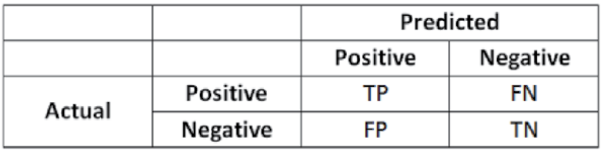
In order to evaluate the performance of the modeling algorithms, appropriate metric is chosen. Based on the research question that the study is going to address, appropriate performance criteria and its measure has to be chosen. Should also consider the ability and feasibility of the work and the study.

Confusion Matrix Confusion Matrix - Confusion matrix is also known as contingency table, provides a overall performance of the classification model. The **Figure 7** shows the format of a confusion matrix.

This confusion matrix provides important elements of the modeling results, they are described below

- True Positives (TP): The number of observations in the positive target class which were correctly classified by the model.

- False Negatives (FN): The number of observations in the positive target class which were incorrectly classified as in the negative target class by the model.



		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Figure 7.
Confusion matrix.

- False Positives (FP): The number of observations in the negative target class which were incorrectly classified as in the positive target class by the model.

- True Negatives (TN): The number of observations in the negative target class which were correctly classified by the model.

HAR is a Multilablel classification problem. The main challenge in classification task is to correctly classify the target variables. Only accuracy score cannot give us the overall performance of the model. Hence, confusion matrix which gives the actual number of correct and incorrect predictions made for each target class is considered. Additionally, precison, recall and f1 score is computed. But for comparisions accuracy and f1 score are considered.

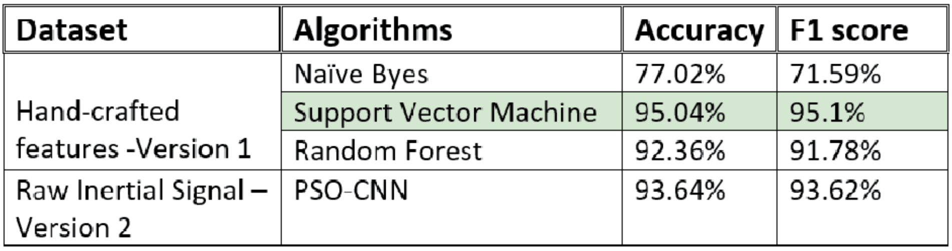
7. Results and discussion

The performance of PSO-CNN is evaluated against Machine Learning and Deep Learning Algorithms.

Comparing PSO-CNN with machine learning algorithms: The analysis of results is performed by comparing PSO-CNN with Machine Learning Algorithms. The below **Figure 8** shows the results.

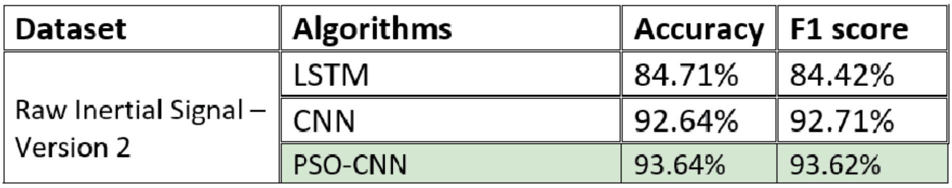
From the table, it is evident that Support Vector Machine achieved accuracy of 95.04% and F1 score 95.1%. The machine learning models were built using Hand-crafted features -Version 1 Dataset. The model achieved satisfactory results without performing any Data Dimensionality reduction techniques. On the other hand, PSO-CNN also achieved considerable results with raw sensor data with accuracy of 93.64%. However, the hand-crafted feature extraction process requires human effort to manually design the features.

Comparing PSO-CNN with Deep learning algorithms: The analysis of results is performed by comparing PSO-CNN with Deep Learning Algorithms. The below **Figure 9** shows the results.



Dataset	Algorithms	Accuracy	F1 score
Hand-crafted features -Version 1	Naïve Byes	77.02%	71.59%
	Support Vector Machine	95.04%	95.1%
	Random Forest	92.36%	91.78%
Raw Inertial Signal – Version 2	PSO-CNN	93.64%	93.62%

Figure 8.
Comparsion of PSO-CNN with machine learning algorithms.



Dataset	Algorithms	Accuracy	F1 score
Raw Inertial Signal – Version 2	LSTM	84.71%	84.42%
	CNN	92.64%	92.71%
	PSO-CNN	93.64%	93.62%

Figure 9.
Comparsion of PSO-CNN with deep learning algorithms.

From the table, it is clear that PSO-CNN was able to achieve high performance of accuracy when compared with LSTM and CNN models. LSTM performance was low with accuracy 84.71% and F1 score with 84.42%. This, PSO-CNN gained better results than the state-of-the art CNN model. For a classification problem, the capability of the modeling algorithm to classify each target class correctly also plays a major role. Each The algorithm's ability to classify each activity like walking, sitting, laying are discussed. From the classification report of PSO-CNN **Figure 9**, it is evident that PSO-CNN was able to classify most number of activities correctly.

7.1 Hypothesis evaluation

Two hypothesis are formed for the evaluation of the experiment.

Hypothesis 1

- Null Hypothesis(H0): If Particle Swarm Optimized Convolutional Neural Network is used instead of supervised machine learning then there is no significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

- Alternate Hypothesis (HA): If Particle Swarm Optimized Convolutional Neural Network is used instead of supervised machine learning, then there is significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

From the **Figure** 5.13 which compares the results of PSO-CNN with machine learning algorithms, it is evident that PSO-CNN did not achieve better results than SVM. Hence, there is no significant evidence to reject the null hypothesis.

Hypothesis 2

- Null Hypothesis(H0): If Particle Swarm Optimized Convolutional Neural Network is used instead of deep learning algorithms, then there is no significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

- Alternate Hypothesis (HA): If Particle Swarm Optimized Convolutional Neural Network is used instead of deep learning algorithms, then there is significant improvement in classification of Human Activity in terms of accuracy and the F1 score.

From the **Figure** 5.14, it is evident that PSO-CNN achieved higher accuracy than deep learning algorithms CNN and LSTM. Hence we have significant evidence to reject the null hypothesis.

8. Conclusion

This section provides a overall review of the current research. Gives a summary of the research overview, problem definition along with key findings in Experiment Design. Suggestions for future work are highlighted.

Research Overview: The main aim of the research was to predict the human activity using the data collected from Inertial sensors. Detailed analysis on the existing research is made which includes the type of data used and approaches used for Human Activity Recognition task. Broadly, HAR task was achieved using two approaches namely machine learning and deep learning by which used hand – crafted features data and raw inertial sensor data, respectively.

There was not much exploration done on the meta-heuristic optimisation algorithms like Particle Swarm Optimization. With this motivation and considering the gaps in research, the current study was aimed to perform the experiments with two versions of the datasets using various algorithms. As a result, the data gathering, and preparation was performed separately for both the datasets. To compare the performance of PSO based CNN, various modeling algorithms were chosen, and was evaluated using the performance metrics.

The research conducted with the objective to find the classifier with high predictive accuracies compared with two different family of modeling algorithms.

Design/Experimentation, Evaluation & Results: CRISP-DM approach was followed through out the project to get the best outcomes at each step. Accordingly, the implementation began with performing Data Gathering, Data Understanding and Data Preparation for both the data sets separately.

The design involved performing experiments with two versions of the data sets. For hand-crafted features, Machine Learning Algorithms were used for modeling. And since, deep neural networks have capability to take the raw input without any domain-knowledge applied, raw inertial signals data was used. Furthermore, the performance of PSO-CNN is evaluated with suitable metric.

The results were tabulated and detailed analysis was given and it is proved that PSO-CNN showed good results than Deep Learning algorithms, but failed to achieve satisfactory results compared to machine learning algorithms.

Contributions and impact:

Detailed literature review was performed emphasizing on the applications of Human Activity Recognition in various fields. In particular, Sensor Based HAR is highlighted for the readers. This also detailed about the current state of the art techniques in HAR.

A systematic investigation is done for importing two versions of the sensor datasets. This can be used as reference for future works.

Illustrated that PSO based CNN proved to be the best classifier for data where human-engineered feature knowledge is not needed.

Additionally, the work tries to enhance the performance of state-of-the-art design of the CNN model by using Optimisation. This adds up to the generalization of using PSO-CNN model for other Activity Recognition tasks.

In the current research PSO algorithm is used to find the optimal architectures in deep convolutional neural network. Furthermore it make use of the benefits of global and local exploration capabilities of the particle swarm optimization technique PSO and the gradient descent back-propogation thereby to form an efficient searching algorithm this is because the performance of deep convolution network extremely depends on their network structure used and hyper-parameter selections.

In order to find the best hyper parameters lot of training time is employed which requires the deep understanding of CNN architecture and also the domain knowledge. Hence PSO-CNN is employed to optimize these parameter configurations and through which efficient parameters are evolved that would increase the performance with less training time.

Future Work and recommendations The current research can be explored and improved in many ways so as to improve the human activity recognition tasks. The proposed approach also provides a flexible methodology where one can change the initial parameter settings of both PSO and CNN. In this way a trade-off between the model generalization capabilities and complexity of the model can be justified. From the experimental results it is illustrated that PSO has been shown to converge faster and find the best configuration with less training time. This exceed he performance of state-of-the-art results obtained in the domain of HAR.

To some extent, the algorithm failed to recognize the similar he activities like WALKING_UPSTAIRS and WALKING, LAYING and SITTING. This may be due to the insufficient data. The solution can be further explored with large time series data.

The experiment is conducted to explore the capability of deep learning algorithms in HAR tasks. In order to generalize the model capability, this can be applied to other Activity Recognition tasks which includes Time Series data.

Acknowledgements

I would first like to express my sincere thanks to my supervisor Prof. Bojan Bozic for his continuous support, guidance and advice throughout the project. It’s been a good experience and honor to work under his supervision.

Abbreviations

HAR	Human Activity Recognition
HAR	Human Activity Recognition
ADL	Activities of Daily Living
CRISP-DM	Cross Industry Standard Process for Data Mining
DT	Decison Tree
ML	Machine Learning
LR	Multinomial Logistic Regression
SVM	Support Vector Machine
HMM	Hidden Markov Model
LSTM	Long Short Term Memory
CNN	Convolutional Neural Network
PSO	Particle Swarm Optimisation
PCA	Principal Component Analysis

Author details

Preethi Gunishetty Devarakonda*† and Bojan Bozic*†
Technological University Dublin, Dublin, Ireland

*Address all correspondence to: preethigd6@gmail.com
and bojan.bozic@tudublin.ie

† These authors contributed equally.

IntechOpen

© 2021 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

References

[1] Aurangzeb K, Haider I, Khan MA, Saba T, Javed K, Iqbal T, et al. Human behavior analysis based on multi-types features fusion and Von Nauman entropy based features reduction. Journal of Medical Imaging and Health Informatics. 2019;9(4):662–669.

[2] Chavarriaga R, Sagha H, Calatroni A, Digumarti ST, Tröster G, Millán JdR, et al. The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition. Pattern Recognition Letters. 2013;34 (15):2033–2042.

[3] Wang J, Chen Y, Hao S, Peng X, Hu L. Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters. 2019;119:3–11.

[4] Piwek L, Ellis DA, Andrews S, Joinson A. The rise of consumer health wearables: promises and barriers. PLoS medicine. 2016;13(2):e1001953.

[5] Cleland I, Kikhia B, Nugent C, Boytsov A, Hallberg J, Synnes K, et al. Optimal placement of accelerometers for the detection of everyday activities. Sensors. 2013;13(7):9183–9200.

[6] Parkka J, Ermes M, Korpipaa P, Mantyjarvi J, Peltola J, Korhonen I. Activity classification using realistic data from wearable sensors. IEEE Transactions on information technology in biomedicine. 2006;10(1):119–128.

[7] Jiang W, Yin Z. Human activity recognition using wearable sensors by deep convolutional neural networks. In: Proceedings of the 23rd ACM international conference on Multimedia; 2015. p. 1307–1310.

[8] Yu H, Cang S, Wang Y. A review of sensor selection, sensor devices and sensor deployment for wearable sensor-based human activity recognition systems. In: 2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA). IEEE; 2016. p. 250–257.

[9] Lara OD, Labrador MA. A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials. 2012;15(3):1192–1209.

[10] Bengio Y. International Conference on Statistical Language and Speech Processing. 2013.

[11] Figo D, Diniz PC, Ferreira DR, Cardoso JM. Preprocessing techniques for context recognition from accelerometer data. Personal and Ubiquitous Computing. 2010;14(7): 645–662.

[12] Khusainov R, Azzi D, Achumba IE, Bersch SD. Real-time human ambulation, activity, and physiological monitoring: Taxonomy of issues, techniques, applications, challenges and limitations. Sensors. 2013;13(10):12852–12902.

[13] Hassan MM, Uddin MZ, Mohamed A, Almogren A. A robust human activity recognition system using smartphone sensors and deep learning. Future Generation Computer Systems. 2018;81:307–313.

[14] Khan AM, Siddiqi MH, Lee SW. Exploratory data analysis of acceleration signals to select light-weight and accurate features for real-time activity recognition on smartphones. Sensors. 2013;13(10):13099–13122.

[15] Ravi N, Dandekar N, Mysore P, Littman ML. Activity recognition from accelerometer data. In: Aaai. vol. 5; 2005. p. 1541–1546.

[16] Yang J. Toward physical activity diary: motion recognition using simple acceleration features with mobile

phones. In: Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics; 2009. p. 1–10.

[17] Kose M, Incel OD, Ersoy C. Online human activity recognition on smart phones. In: Workshop on Mobile Sensing: From Smartphones and Wearables to Big Data. vol. 16; 2012. p. 11–15.

[18] Kwapisz JR, Weiss GM, Moore SA. Activity recognition using cell phone accelerometers. ACM SigKDD Explorations Newsletter. 2011;12(2): 74–82.

[19] Tang Y, Teng Q, Zhang L, Min F, He J. Efficient convolutional neural networks with smaller filters for human activity recognition using wearable sensors. arXiv preprint arXiv: 200503948. 2020.

[20] Shaafi A, Salem O, Mehaoua A. Improving Human Activity Recognition Algorithms using Wireless Body Sensors and SVM. In: 2020 International Wireless Communications and Mobile Computing (IWCMC). IEEE; 2020. p. 607–612.

[21] Aslan MF, Durdu A, Sabanci K. Human action recognition with bag of visual words using different machine learning methods and hyperparameter optimization. Neural Computing and Applications. 2020;32(12):8585–8597.

[22] Arel I, Rose DC, Karnowski TP. Deep machine learning-a new frontier in artificial intelligence research [research frontier]. IEEE computational intelligence magazine. 2010;5(4):13–18.

[23] Najafabadi M, Villanustre F, Khoshgoftaar T, Seliya N, Wald R, Muharemagic E. In: Deep Learning Techniques in Big Data Analytics; 2016. p. 133–156.

[24] Plötz T, Hammerla N, Olivier P. Feature learning for activity recognition in ubiquitous computing; 2011.

[25] Liu Z, Wu M, Zhu K, Zhang L. SenSafe: A smartphone-based traffic safety framework by sensing vehicle and pedestrian behaviors. Mobile Information Systems. 2016;2016.

[26] Guan Y, Plötz T. Ensembles of deep lstm learners for activity recognition using wearables. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies. 2017;1(2): 1–28.

[27] Vepakomma P, De D, Das SK, Bhansali S. A-Wristocracy: Deep learning on wrist-worn sensing for recognition of user complex activities. In: 2015 IEEE 12th International conference on wearable and implantable body sensor networks (BSN). IEEE; 2015. p. 1–6.

[28] Walse KH, Dharaskar RV, Thakare VM. Pca based optimal ann classifiers for human activity recognition using mobile sensors data. In: Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1. Springer; 2016. p. 429–436.

[29] Edel M, Köppe E. Binarized-blstm-rnn based human activity recognition. In: 2016 International conference on indoor positioning and indoor navigation (IPIN). IEEE; 2016. p. 1–7.

[30] Inoue M, Inoue S, Nishida T. Deep recurrent neural network for mobile human activity recognition with high throughput. Artificial Life and Robotics. 2018;23(2):173–185.

[31] Ha S, Yun JM, Choi S. Multi-modal convolutional neural networks for activity recognition. In: 2015 IEEE International conference on systems, man, and cybernetics. IEEE; 2015. p. 3017–3022.

[32] Singh MS, Pondenkandath V, Zhou B, Lukowicz P, Liwickit M.

Transforming sensor data to the image domain for deep learning—An application to footstep detection. In: 2017 International Joint Conference on Neural Networks (IJCNN). IEEE; 2017. p. 2665–2672.

[33] Li X, Zhang Y, Marsic I, Sarcevic A, Burd RS. Deep learning for rfid-based activity recognition. In: Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM; 2016. p. 164–175.

[34] Kennedy J. Swarm intelligence. In: Handbook of nature-inspired and innovative computing. Springer; 2006. p. 187–219.

[35] Godino-Llorente JI, Gomez-Vilda P. Automatic detection of voice impairments by means of short-term cepstral parameters and neural network based detectors. IEEE Transactions on Biomedical Engineering. 2004;51(2): 380–384.

[36] Mihanpour A, Rashti MJ, Alavi SE. Human Action Recognition in Video Using DB-LSTM and ResNet. In: 2020 6th International Conference on Web Research (ICWR). IEEE; 2020. p. 133–138.

[37] Qassim H, Verma A, Feinzimer D. Compressed residual-VGG16 CNN model for big data places image recognition. In: 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). IEEE; 2018. p. 169–175.

[38] Iandola F, Moskewicz M, Karayev S, Girshick R, Darrell T, Keutzer K. Densenet: Implementing efficient convnet descriptor pyramids. arXiv preprint arXiv:14041869. 2014.

[39] Gudise VG, Venayagamoorthy GK. Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In: Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706); 2003. p. 110–117.

[40] Carvalho M, Ludermir TB. Particle swarm optimization of feed-forward neural networks with weight decay. In: 2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06). IEEE; 2006. p. 5–5.

[41] Dehuri S, Roy R, Cho SB, Ghosh A. An improved swarm optimized functional link artificial neural network (ISO-FLANN) for classification. Journal of Systems and Software. 2012;85(6): 1333–1345.

[42] Junior FEF, Yen GG. Particle swarm optimization of deep neural networks architectures for image classification. Swarm and Evolutionary Computation. 2019;49:62–74.

Chapter 5

Particle Swarm Optimization Algorithms with Applications to Wave Scattering Problems

Alkmini Michaloglou and Nikolaos L. Tsitsas

Abstract

Particle Swarm Optimization (PSO) algorithms are widely used in a plethora of optimization problems. In this chapter, we focus on applications of PSO algorithms to optimization problems arising in the theory of wave scattering by inhomogeneous media. More precisely, we consider scattering problems concerning the excitation of a layered spherical medium by an external dipole. The goal is to optimize the physical and geometrical parameters of the medium's internal composition for varying numbers of layers (spherical shells) so that the core of the medium is substantially cloaked. For the solution of the associated optimization problem, PSO algorithms have been specifically applied to effectively search for optimal solutions corresponding to realizable parameters values. We performed rounds of simulations for the the basic version of the original PSO algorithm, as well as a newer variant of the Accelerated PSO (known as "Chaos Enhanced APSO"/ "Chaotic APSO"). Feasible solutions were found leading to significantly reduced values of the employed objective function, which is the normalized total scattering cross section of the layered medium. Remarks regarding the differences and particularities among the different PSO algorithms as well as the fine-tuning of their parameters are also pointed out.

Keywords: Swarm Intelligence, optimization, particle swarm optimization (PSO), accelerated particle swarm optimization (APSO), chaos-enhanced APSO, chaotic APSO (CAPSO), wave scattering, cloaking

1. Introduction

Particle Swarm Optimization (PSO) is a population-based, stochastic optimization algorithm. It is modelled after the intelligent behavior patterns found in swarms of animals when they manage their biological needs. It was first introduced in 1995 [1], and since then many enhancements and new versions of the algorithm have appeared. The model originates from the behavior of flocks (swarms) of birds when in search of food sources. It was inspired by research carried out by Heppner and Grenander [2], in order to experiment on a "cornfield model". Exploiting these studies, Kennedy and Eberhart developed the PSO algorithm, in which the members of the swarm, called *particles* have some form of memory and common knowledge and are motivated by a common goal; in the mathematical framework this goal is the global optimum of the objective function of the optimization problem. The particles' positions represent the solutions, and depending on the method, they can also have velocity or other characteristics, or even a societal structure. The swarm acts in alliance, aims to be effective, and there exists enough individuality to achieve diversity in possible solutions. By design, particle swarm optimization is inseparable from Swarm Intelligence. The swarm, as defined in literature, is designed to follow the basic principles of Swarm Intelligence, namely *proximity*, *quality*, *diverse response*, *stability* and *adaptability*.

In this chapter, two PSO algorithms are presented. First, the original PSO, which utilizes a global best position $\mathbf{g}^*$ and an individual best position $\mathbf{x}^*$ for the particles, which are described by both their position and velocity. This is considered to be the basic PSO algorithm, and the version chosen [3] also utilizes an inertia mechanism to describe the particles' movement. The second algorithm is an enhancement of the Accelerated Particle Swarm Optimization (APSO) algorithm, referred to as the Chaotic APSO (CAPSO) [4]. In this algorithm, the particles update their position in a single step and are only described by position, not velocity vectors. Additionally, they only use the global best position $\mathbf{g}^*$ as an attraction to the optimum. Specified parameters get updated to fine tune the process, and precisely, the attraction parameter β updates through the use of chaotic maps.

Both aforementioned algorithms have been applied to wave scattering problems, and results of numerical implementations alongside with conclusions are provided. Precisely, we consider the cloaking problem concerning the excitation of a layered spherical medium with perfect electric conducting (PEC) core by an external dipole. The main purpose is to determine suitable parameters of the magneto-dielectric layers covering the PEC core so that the scattered far-field is significantly reduced for a wide range of observation angles. Obtained optimal designs demonstrating efficient cloaking performance are presented exhibiting reduced values of the bistatic scattering cross section for realizable coatings parameters. It is particularly stressed that the CAPSO determines optimal values of the scattering problem's variables, which yield highly-efficient cloaking designs by employing ordinary coatings materials.

PSO algorithms in computational methodologies and engineering applications involving electromagnetic waves were initially developed in [5, 6], where implementations in antenna design were also proposed. A quantum PSO algorithm, based on Quantum Mechanics rather than the Newtonian rules considered in the original versions of the algorithm, was developed in [7] and applied for finding a set of infinitesimal dipoles producing the same near and far fields of a circular dielectric resonator antenna. A molecular dynamics formulation of the PSO algorithm leading to a physical theory for the swarm environment was presented in [8] and applied to problems of synthesis of linear array antennas. Variants of PSO algorithms with relevant applications in electromagnetic design problems, like microwave absorbers and base station antenna optimization for mobile communications were analyzed in [9]. Specifically, concerning the cloaking behavior of layered media, related optimization problems were investigated in [10–16]. Optimization techniques for metadevices design are overviewed in [17].

2. Particle Swarm Optimization (PSO)

In this section, the basic principles of Particle Swarm Optimization (PSO) are presented and an in depth description of the algorithms that have been developed and applied for the considered cloaking problems is given. After discussing the theoretical basis of the swarm optimization method and its ties to Swarm Intelligence, the PSO algorithm and the chaotic-enhanced version (CAPSO) of the accelerated particle swarm optimization (APSO) algorithm are described.

2.1 Introduction to PSO

PSO is a population-based stochastic optimization algorithm, modelled after the behavior of swarms of animals, like flocks of birds, swarms of various types of insects or ants or school of fish [18]. In literature, it is also categorized as a metaheuristic algorithm. Usually, the population is referred to as a swarm. These types of methods are also considered to be and referred to as behaviorally-inspired, opposed to evolutionary-based methods like genetic algorithms, although some parallels can be drawn between them, with regards to their inner workings. Another similar research field is artificial life. The term, as well as the algorithm, was originally proposed in 1995 [1] and although PSO's precursor was the study and simulation of animal behavior (even in the hopes of studying human social behavior), it grew into an optimizer, with a simple, yet well-defined description. By definition, PSO is indissolubly linked to Swarm Intelligence.

The appeal of swarm optimizers is due to numerous reasons. There exist many types of biological swarms, so one can safely assume that they constitute a promising pool of inspiration and resources to draw methods and conclusions from. The global adaptive behavior of the swarm, and its co-operational behavior and decision making, is practical but not strictly utilitarian, since a swarm behaves with fluid and elegant coordination. Additionally, the way a biological swarm acts can be clearly and directly perceived by humans. Thus, we have a better understanding of the animals' purpose, goals, communication and utility unlike other natural phenomena, which can be way more abstract, complicating the creation of a well-structured model or method.

Since the initial introduction of PSO, several variations of the method have been introduced. A plethora of algorithms have been and are still being designed with different parameters and applications in mind, in order to adjust to specific problems. These numerous variants are widely used and examined, and, thus, PSO has grown to be a very effective technique. In the following subsection, a more generic description of the swarm and its behavior is presented, while detailed descriptions of specific algorithms are given in the sequel.

2.1.1 The Particle Swarm

The term "particle" refers to the points in the n-dimensional space (where n is the number of variables of the objective function) which represent the biological entities of the swarm. Let us assume that the representative animal species is birds. The swarm consists of the entirety of the particles, making up the population. The particles have neither mass, nor volume and although they could be considered points in space, the term particle has been chosen as a good compromise, due to its more active usage in literature [1].

Each particle maintains information about two characteristics; its position $\mathbf{x}$ and velocity $\mathbf{u}$. The position is strictly the most important characteristic, since it represents the solutions to the objective function of the optimization problem. The particles also have some common memory of useful information, since they share information regarding the best position the swarm has achieved (based on the objective function), referred to as the "global best" $\mathbf{g}^*$. In nature, this knowledge could refer to food, shelter or destination. Depending on the variant or type of PSO algorithm being used, they can also remember their individual best position $\mathbf{x}^*$, or a set of best positions if they follow a different type of structure, or even a best position that represents their social clan and/or leader.

According to [19], the biological swarm has three specific qualities. First, *cohesiveness*: the members are not unrelated to each other and all of them are part of the same group, thus to an extent, they “stick together”. Second, there is *separation*, the members actively try to not collide with each other and move with some respect to the average distance between them. Last, there is *alignment*, the whole population actively tries to move towards the same direction as a group effort. In Biology, this is the source of food, while in Optimization it is the optimum of the problem. Of course, since particles are designed to be without mass and volume, separation is not a physical quality the swarm is forced to have. When converging to a solution, all particles end at or near to the specific position representing this solution. However, separation exists as a principle, since particle “collision” does not hinder their movement in any way, shape or form. Particles are separate entities to each other to a certain degree since they are created with their individual attributes (e.g. initial positions, individual best, clan leader and more, depending on the algorithm) and act accordingly, having a degree of autonomy, while searching in unison with respect to the swarm.

2.1.2 Basic Principles of Swarm Intelligence

In order to clearly establish the link between PSO and Swarm Intelligence, we present a comprehensible list of Swarm Intelligence principles, in reference to Millonas’ categorization [1, 18, 20]. Let us refer to a group of entities that collectively act and behave. This group has Swarm Intelligence if these principles are true.

1. **Proximity principle**. The members of the group should be able to handle and do elementary space and time computations. This means that the group can behaviorally respond to environmental stimuli and changes. Also, they should be able to do so in order to better conduct their main utilities and functions which are specific to this group. Such activities vary, depending on the group, for example a swarm of ants could have a main utility of food foraging.

2. **Quality principle**. The group should not only react to time and space stimuli, but also check for quality factors and parameters, e.g. safety.

3. **Principle of diverse response**. The group should not respond to its environment in an absolutely ordered manner. There should be safety locks, and insurance policies for it to survive in case of unpredicted changes and fluctuations in the environment. Resources should not all rely to a single point of focus. Therefore, the swarm must be prepared to act and respond with diverse and alternative solutions.

4. **Principle of stability**. The group as a whole, should not reform its behavior patterns into a completely alternate mode every time a change happens, since such an intense structural and behavioral change wastes too much energy, and might eliminate the possibility of reaching good results.

5. **Principle of adaptability**. However, the group should also be able to switch its behavioral mode, provided this change is a positive one and the group has ways of knowing so.

One can observe that stability and adaptability are principles that go hand-in-hand and the best strategy to approach, is to safely explore a viable middle ground. Some level of randomness or noise should exist in the group, to a degree that diverse response is allowed to happen. That is the reason why such parameters are usually very important to the algorithms and can dramatically change their results.

PSO dictates that the swarm acts in a way which is complicit with the aforementioned principles. In the original PSO publication, Kennedy and Eberhart do confirm that the PSO algorithm has been designed to function in this manner. Similar explanations and proofs were provided in literature [1, 18]. As it has been briefly mentioned, in PSO, particles maintain their position and velocity, and have the ability to react to environmental time and space stimuli in order to update them. They do so in time steps-iterations, thus following the *proximity principle*. The swarm reacts to the global best value $\mathbf{g}^*$ alongside with other quality factors when doing said updates, so it enforces the *quality principle*. Said quality factors, do not prevent the *diverse response*, because the swarm avoids behaving in an excessively restricted manner. This is encouraged by diversity and noise existing within the swarm. Lastly, the swarm bases its behavioral change(s) on a well-defined criterion (which includes the global best position $\mathbf{g}^*$), thus providing *adaptability* without jeopardizing the swarm's *stability*. The mode of behavior changes when it is beneficial and cost-effective.

2.2 The PSO algorithm

In this section, we refer to the original PSO algorithm [1], alongside with the upgrade proposed in 1998 [3] which utilizes an inertia mechanism.

2.2.1 Description

The PSO algorithm follows all the principles and characteristics mentioned so far. By default, a maximization optimizer is considered due to the way the model works, but there exist methods to effectively utilize the algorithm in order to find minima as well.

The behavior of the flock was heavily inspired by and based upon Heppner's [2] simulation of a bird flock, referred to as a *cornfield model* or *cornfield vector*. Heppner wanted to simulate the way a flock of birds moves while searching for food (namely "cornfield" in the simulation). The birds' behavior in real life, hints to the existence of what we refer to as a common sense or knowledge, meaning that members of the flock have the ability to share knowledge originating from their peers without having experienced it themselves. This serves as both a cognitive function and a means of communication. Very often, we do witness this phenomenon; flocks of birds can discover a new bird feeder in their area in a matter of few hours, and an increasing number of them will systematically start visiting it. This behavior was modelled in the simulation, in which the birds were given two types of memory. For the flock's memory of food sources they were given what we previously referred to as the global best $\mathbf{g}^*$ and for their individual memory, they kept information of the best position they have individually visited, their $\mathbf{x}^*$. There were also extra parameters to adjust how effectively each memory spot affects the birds' movement and behavior.

Kennedy and Eberhart [1] utilized Heppner's simulation model, and designed the PSO algorithm in order to use these advantageous observations. So, in the PSO algorithm, the model is as follows.

1. When particles locate a good solution to the optimization problem, this knowledge is transmitted to the whole swarm, meaning that the $\mathbf{g}^*$ value is known to each member.

2. All particles do gravitate towards good solutions, but not in an absolute forced way, because,

3. all particles maintain their personal memory spot for their own value $\mathbf{x}^*$, thus preserving some ability for independent thinking.

The particles move with respect to Newton's laws of motion, while there exist parameters to insert some randomness. There exist also learning rates that the particles adhere to.

In 1998, Shi and Eberhart [3] proposed strategies on how to fine-tune the parameters of the original PSO algorithm. Particularly, they suggested the use of an inertia weight mechanism θ applied to the particles' movement because it was found in experimentation that the particle velocities built up too fast and the maximum of the objective function can be skipped. Usually, the inertia decreases in a linear manner while the iterations of the algorithm run, and it gets updated once per iteration i. For the inertia, the values $\theta_{\max} = 0.9$ and $\theta_{\min} = 0.4$ are commonly used [19].

$$\theta_i = \theta_{\max} - \frac{\theta_{\max} - \theta_{\min}}{i_{\max}} i \tag{1}$$

Therefore, the velocity and position updates are described, respectively, in the following formulae, with respect to iteration i:

$$\mathbf{u}_i = \theta_i \mathbf{u}_{i-1} + c_1 r_1 [\mathbf{x}^* - \mathbf{x}_{i-1}] + c_2 r_2 [\mathbf{g}^* - \mathbf{x}_{i-1}] \tag{2}$$

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{u}_i, \tag{3}$$

where the parameters c_1 and c_2 are the cognitive (individual) and social (group) learning rates and are usually assumed to both be 2, so that the particle overflies the target approximately half of the time. It is interesting to note that if c_1 and c_2 are different to each other, then the particles will in time favor one type of best position (or *behavior*) over the other. In a way, this would conceptually translate to the particles choosing to be more selfish than social and vice versa. This could lead to less optimal solutions than the ones expected. The parameters r_1 and r_2 are uniformly distributed random numbers in the range from 0 to 1.

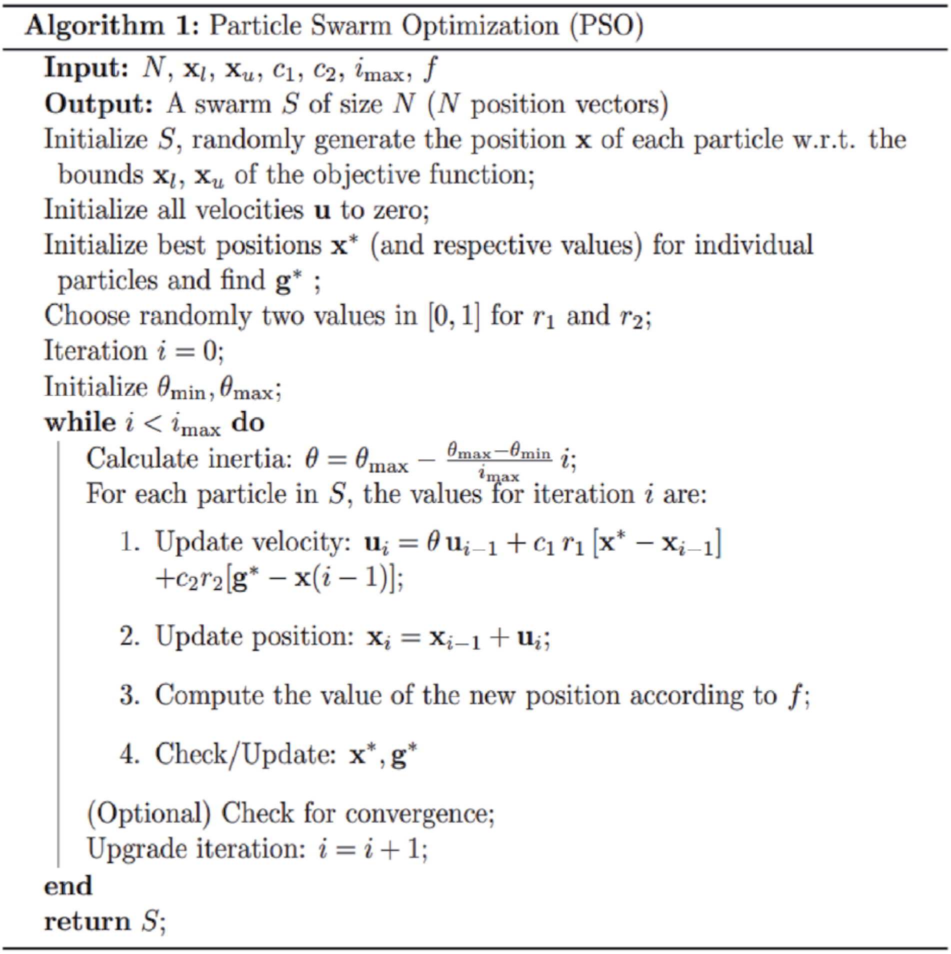
2.2.2 Algorithm

After describing the model of the algorithm, a concrete and defined algorithm can be presented for the computational implementation. The algorithm is depicted in pseudo code form in **Figure 1**.

Regarding the various parameters, we make the following remarks. Usually a size of 20 to 30 for N is assumed, but these numbers can vary depending on the optimization problem. The bigger the swarm, the more evaluations of the objective function f are made during each iteration, thus due to the computations, the algorithm becomes more time consuming. From a programmer's point of view, f does not necessarily need to be an input, however, it is depicted in this manner for reasons of clarity.

2.3 The CAPSO algorithm

As we have previously mentioned, in the original version of the PSO algorithm, both a global ($\mathbf{g}^*$) and an individual best ($\mathbf{x}^*$) are used, with the particles' position being greatly affected by them. The accelerated particle swarm optimization algorithm (APSO) however, introduced by Yang [21], follows a different approach. The



```
Algorithm 1: Particle Swarm Optimization (PSO)
  Input: $N$, $\mathbf{x}_l$, $\mathbf{x}_u$, $c_1$, $c_2$, $i_{\max}$, $f$
  Output: A swarm $S$ of size $N$ ($N$ position vectors)
  Initialize $S$, randomly generate the position $\mathbf{x}$ of each particle w.r.t. the
    bounds $\mathbf{x}_l$, $\mathbf{x}_u$ of the objective function;
  Initialize all velocities $\mathbf{u}$ to zero;
  Initialize best positions $\mathbf{x}^*$ (and respective values) for individual
    particles and find $\mathbf{g}^*$ ;
  Choose randomly two values in $[0, 1]$ for $r_1$ and $r_2$;
  Iteration $i = 0$;
  Initialize $\theta_{\min}, \theta_{\max}$;
  while $i < i_{\max}$ do
      Calculate inertia: $\theta = \theta_{\max} - \frac{\theta_{\max} - \theta_{\min}}{i_{\max}} i$;
      For each particle in $S$, the values for iteration $i$ are:
        1. Update velocity: $\mathbf{u}_i = \theta\, \mathbf{u}_{i-1} + c_1 r_1 [\mathbf{x}^* - \mathbf{x}_{i-1}]$
           $+ c_2 r_2 [\mathbf{g}^* - \mathbf{x}(i-1)]$;
        2. Update position: $\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{u}_i$;
        3. Compute the value of the new position according to $f$;
        4. Check/Update: $\mathbf{x}^*, \mathbf{g}^*$
      (Optional) Check for convergence;
      Upgrade iteration: $i = i + 1$;
  end
  return $S$;
```

Figure 1.
The PSO algorithm pseudo code.

chaos-enhanced particle swarm optimization, or chaotic APSO (CAPSO) is a variation of the APSO algorithm.

2.3.1 Accelerated Particle Swarm Optimization

It is noted that the individual best $\mathbf{x}^*$ in PSO, acts as a creator of diversity in the swarm. That is not necessarily the only purpose of the individual best, but it is a very prominent one. Thus, this diversity could be recreated by utilizing randomness to bypass the use of the individual best. There exist some algorithms that belong in this more "simplistic" philosophy, and try to use only the most necessary parameters and formulae. The accelerated particle swarm optimization algorithm (APSO), follows this route. APSO has been applied in many optimization problems and is a solid method with good results. One can safely develop and use APSO, and similar methods or variants, while keeping in mind that PSO, or even more its standard versions, is still *in general* a better option if the optimization problem of interest is highly nonlinear and multimodal [21].

Ergo, the APSO algorithm only uses the global best $\mathbf{g}^*$ to generate the velocity vector $\mathbf{u}$, resulting to using a simpler mathematical formula. For a specific particle, during the $i - th$ iteration, the velocity is:

$$\mathbf{u}_i = \mathbf{u}_{i-1} + \alpha(r - 1/2) + \beta(\mathbf{g}^* - \mathbf{x}_{i-1}) \tag{4}$$

where r is a random variable with values from 0 to 1, and the 1/2 is used as a means of convenience. It is suggested [21], that a normal distribution αr_i is used, where r is drawn from N(0,1). Thus, velocity and positions updates are given, respectively, by

$$\mathbf{u}_i = \mathbf{u}_{i-1} + \beta(\mathbf{g}^* - \mathbf{x}_{i-1}) + \alpha r_{i-1}, \tag{5}$$

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{u}_i \tag{6}$$

In [21], the following simplified formula is also suggested for the particle location update in a single step:

$$\mathbf{x}_i = (1 - \beta)\mathbf{x}_{i-1} + \beta\mathbf{g}^* + \alpha r_{i-1}, \tag{7}$$

hence there is no need of utilizing structs or vectors for the velocity, while separate initializations and updates are also avoided.

The typical parameter values for this accelerated PSO are $\alpha \in [0.1, 0.4]$ and $\beta \in [0.1, 0.7]$. More generally, we must keep in mind that these parameters should scale with respect to the scales of the problem variables. A further improvement to APSO [21] is to reduce the randomness as iterations proceed. This means that we can use a monotonically decreasing function specifically for the parameter α, e.g.

$$\alpha = \alpha_0 \gamma^t, (0 < \gamma < 1) \tag{8}$$

or

$$\alpha = \alpha_0 e^{-\gamma t}. \tag{9}$$

Other non-increasing functions $\alpha(t)$ can be used like the example provided in code in [21].

2.3.2 Chaos-Enhanced APSO

Gandomi et al. proposed a variation of the APSO algorithm, the chaotic APSO (CAPSO) [4]. According to the study, the attraction parameter β in (Eq. (7)) is crucially important in determining the speed of the convergence and how the algorithm behaves, since this parameter characterizes the variations of the global best attraction. A well tuned β is of great importance. After parametric investigations, it is suggested that β should be in $[0.2, 0.7]$ for most problems solved by APSO. Additionally, it is noted that the parameter β has no practical reason of remaining a constant. On the contrary, a varying β can offer an advantage in terms of convergence speed and algorithm behavior.

The method suggested for tuning the parameter β is *chaotic maps*. In Mathematics, chaotic maps are evolutionary functions that exhibit some sort of chaotic behavior [22]. Chaotic maps often occur in the study of dynamical systems. Also, they are used to generate fractals. They can change in time in a continuous or discrete manner, but usually chaotic maps are discrete ones. Therefore, they take the form of iterated functions. Chaotic maps are normalized, their variations are always between $[0, 1]$, so they can safely be used for tuning the parameter β.

In the original proposal of CAPSO [4], many chaotic maps were tested in terms of convergence and effectiveness. The results were listed in detail, and it was noted that the *Sinusoidal* map was the best performing one, and the *Singer* map was the second best. Consequently, the Sinusoidal map is the best choice for applications. It was noted that chaotic maps with a unimode centered around their middle tend to produce better results, and Sinusoidal and Singer maps fall into this category. They are as follows:

Sinusoidal Map:

$$x_{k+1} = a{x_k}^2 \sin\left(\pi x_k\right) \tag{10}$$

As an alternative, the following simplified form has also been suggested and applied [4, 23]:

$$x_{k+1} = \sin\left(\pi x_k\right) \tag{11}$$

Singer Map:

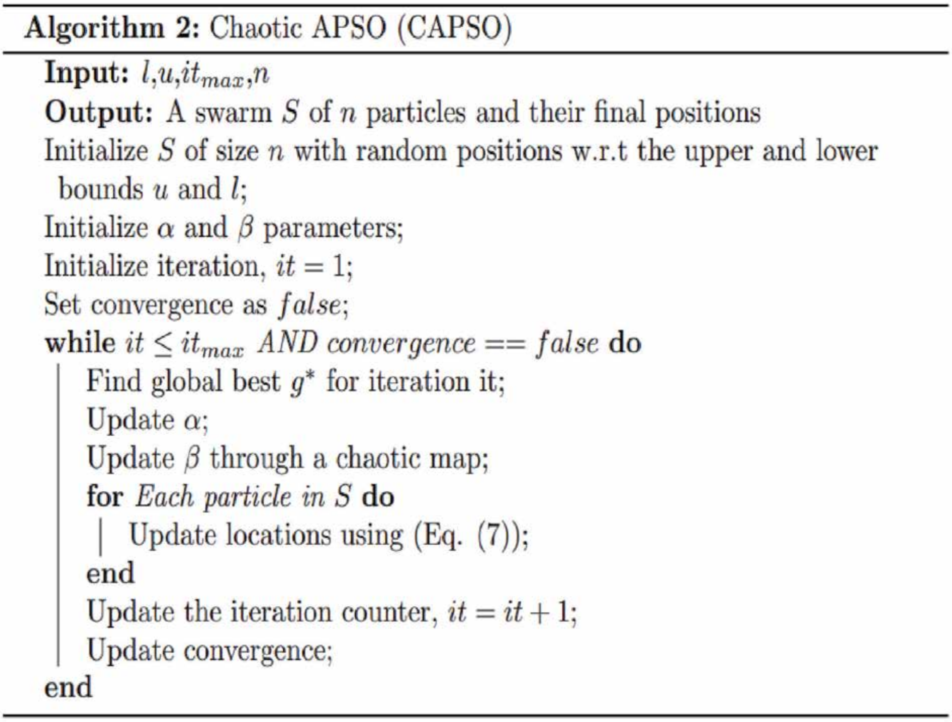
$$x_{k+1} = \mu\left(7.86x_k - 23.31{x_k}^2 + 28.75{x_k}^3 - 13.302875{x_k}^4\right), \tag{12}$$

where $\mu \in [0.9, 1.08]$.

2.3.3 The CAPSO Algorithm

Having described the basis of the APSO algorithm, as well as the improvements added from chaotic maps, the CAPSO algorithm is now presented in pseudo code form in **Figure 2**.

The following information is provided for the various paramaters. Usually a size of 40 for N is considered sufficient, but these numbers can vary depending on the optimization problem. The parameter α gets updated through a chosen $\alpha(t)$ (which



```
Algorithm 2: Chaotic APSO (CAPSO)
Input: $l$,$u$,$it_{max}$,$n$
Output: A swarm $S$ of $n$ particles and their final positions
Initialize $S$ of size $n$ with random positions w.r.t the upper and lower
  bounds $u$ and $l$;
Initialize $\alpha$ and $\beta$ parameters;
Initialize iteration, $it = 1$;
Set convergence as $false$;
while $it \leq it_{max}$ AND convergence == $false$ do
    Find global best $g^*$ for iteration it;
    Update $\alpha$;
    Update $\beta$ through a chaotic map;
    for Each particle in $S$ do
        Update locations using (Eq. (7));
    end
    Update the iteration counter, $it = it + 1$;
    Update convergence;
end
```

Figure 2.
The CAPSO algorithm pseudo code.

is a monotonically decreasing function or a non-increasing function in general). For α, the initial value depends on the scale of the problem variables and on $\alpha(t)$. One can apply the values proposed for APSO, or alternatively $\alpha = 10$ can be chosen for an initial value as a starting point. Testing with different initial values is encouraged. The parameter β is updated through a chaotic map, preferably the Sinusoidal map. In the original paper [4], the maximum iteration number is suggested to be 250. One must keep in mind that depending on the problem, these values might have to be re-evaluated and re-adjusted.

2.4 Development suggestions

Many suggestions can be made regarding the robustness of algorithms, as well as the speed, effectiveness and organization of the code. All these highly depend on the programming language, development technique, programmer expertise, computational load of the optimization problem and numerous more parameters. When developing these algorithms, we must take into consideration all of the above, and more, since applications can greatly diversify from one another.

Below, two suggestions are made regarding the PSO and APSO/CAPSO algorithms, which, when applied, improved the testing process on a complicated wave scattering optimization problem detailed below. However, they are not heavily dependent on the nature of said optimization problem, and they could be proven to be helpful regardless.

1. **Application of constraints/bounds.** A method that reassures that the variables remain in their allowed bounds is vital. This is very common in optimization. If a variable crosses a bound, the lower or higher permitted value can be enforced, with respect to which bound was crossed. This reassures that the swarm will not go out of bounds if it gets driven to do so by a nearby invalid optimum. Additionally, it ensures that the final output of the algorithm is a valid and applicable one, even if it is not the best optimum. For complex optimization problems, constraint/bound checking can be complicated, if for example the variables have to follow specific rules, or have specific characteristics in relation to each other. We can see this technique being applied in APSO's code [21].

2. **Convergence checking.** By default, in most PSO related algorithms, it is implied that the algorithm stops when it reaches a pre-defined maximum of iterations. However, many times, the swarm can find a solution faster than that. Thus, if there is a convergence criterion (representing the degree in which the population agrees on a solution), it can be applied as an end condition for the algorithm. For example, a very common convergence criterion is standard deviation.

3. Particle swarm optimization in wave scattering problems

In this section, PSO optimizations to representative applications of wave scattering theory are presented. Precisely, we investigate the electromagnetic cloaking of spherically layered media excited by an external source. The optimizations concern the determinations of the physical (material) and geometrical characteristics of the layered medium so that the scattered far field generated by the layered medium is significantly reduced.

The scattering geometry is depicted in **Figure 3**. It consists of a layered spherical medium V with external radius a_1. The interior of V is divided by $P-1$ concentric spherical interfaces $r=a_p$ $(p=2,\ldots,P)$ into $P-1$ homogeneous magneto-dielectric layers V_p $(p=1,\ldots,P-1)$, consisting of materials with real relative dielectric permittivities ε_p and magnetic permeabilities μ_p, and surrounding a perfect electric conducting (PEC) core (layer V_P). The exterior V_0 of V has permittivity ε_0, permeability μ_0, and wavenumber k_0. Medium V is excited by an external magnetic dipole, with position vector $\mathbf{r}_0$ on the z-axis and dipole moment along the direction $\hat{\mathbf{y}}$.

The exact solution of the considered scattering problem was determined in [24–26] by means of a combined Sommerfeld and T-matrix methodology in conjunction with suitable eigenfunctions expansions. Specifically, the electric fields in each spherical shell are decomposed into primary and secondary components, which are then expressed as series of the spherical vector wave functions. The unknown coefficients in the expansions of the secondary fields are determined analytically by imposing the transmission boundary conditions on the interfaces of the spherical shells and applying a T-matrix method. It is emphasized that the exact solution of the scattering problem (here this is obtained in the form of a Mie series) is crucial for the fast and efficient implementation of the PSO algorithm in the present setting.

By applying the above-described methodology, we obtain the following expression of the total scattering cross section

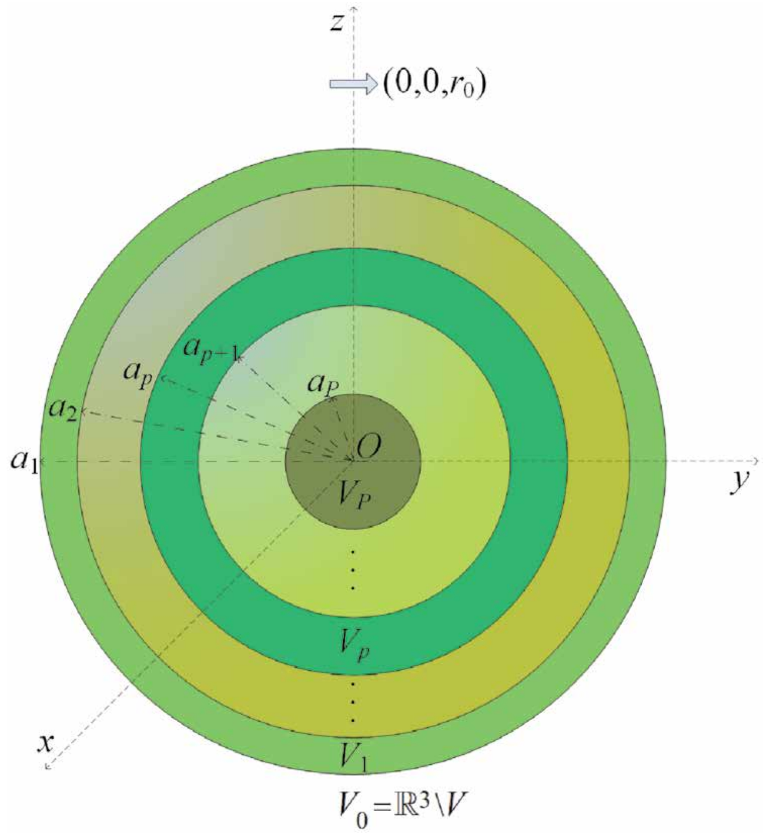


Figure 3.
Geometrical configuration of the considered spherically-layered medium excited by an external dipole.

$$\begin{aligned}\sigma^{\mathrm{t}}(\mathbf{r}_0) &= \frac{1}{4\pi}\int_{S^2}\sigma(\theta,\phi;\mathbf{r}_0)\mathrm{d}s(\hat{\mathbf{r}}) \\ &= \frac{2\pi}{k_0^2}\sum_{n=1}^{\infty}(2n+1)\left[|\gamma_n|^2+|\delta_n|^2\right],\end{aligned} \tag{13}$$

where S^2 denotes the unit sphere in $\mathbb{R}^3$, and $\sigma(\theta,\phi;\mathbf{r}_0)$ is the bistatic (differential) scattering cross section given by

$$\sigma(\theta,\phi;\mathbf{r}_0) = \frac{4\pi}{k_0^2}\left[|S_\theta(\theta;\mathbf{r}_0)|^2\cos^2\phi + |S_\phi(\theta;\mathbf{r}_0)|^2\sin^2\phi\right], \tag{14}$$

while functions $S_\theta(\theta;\mathbf{r}_0)$ and $S_\phi(\theta;\mathbf{r}_0)$ are defined by

$$S_\theta(\theta;\mathbf{r}_0) = \sum_{n=1}^{\infty}\frac{(-1)^n(2n+1)}{\sqrt{n(n+1)}}\left[\delta_n\frac{P_n^1(\cos\theta)}{\sin\theta} - \gamma_n\frac{\partial P_n^1(\cos\theta)}{\partial\theta}\right], \tag{15}$$

$$S_\phi(\theta;\mathbf{r}_0) = \sum_{n=1}^{\infty}\frac{(-1)^n(2n+1)}{\sqrt{n(n+1)}}\left[\gamma_n\frac{P_n^1(\cos\theta)}{\sin\theta} - \delta_n\frac{\partial P_n^1(\cos\theta)}{\partial\theta}\right], \tag{16}$$

with P_n^1 the first-order Legendre function of degree n, and

$$\gamma_n = \frac{h_n(k_0r_0)}{h_0(k_0r_0)}\mathrm{i}^n\alpha_n, \quad \delta_n = \frac{\hat{h}_n'(k_0r_0)}{\hat{h}_0(k_0r_0)}\mathrm{i}^{n-1}\beta_n, \tag{17}$$

where h_n is the spherical Hankel function of order n, and $\hat{h}_n(z) = zh_n(z)$. The coefficients α_n and β_n are defined in [24].

The objective function we consider in the optimization schemes is the *normalized total scattering cross section* $\sigma^{\mathrm{t}}(\mathbf{r}_0)/(\pi a_{\mathrm{PEC}}^2)$, where a_{PEC} is the radius of the PEC sphere to be cloaked when covered by suitable coating magneto-dielectric layers. Achieving small values of this objective function provides efficient designs in terms of significant reductions in the scattered far-field. In [27], the backscattering cross section $\sigma(\theta,0;\mathbf{r}_0)$ was used as the objective function. The latter can yield efficient designs only in traditional monostatic scenarios, while the present consideration of the total scattering cross section as the objective function shows the actual scattered far-field's characteristics for all observation angles.

For the numerical solution of the scattering problem, we used the code developed in [24], which is valid for an arbitrary number P of layers. The above-described PSO algorithms were implemented in MATLAB®. The swarms were MATLAB structs or arrays for which we followed the steps of Algorithms 1 or 2 presented above. The components of the position vector consisted of the optimization variables a_p of the radii, ε_p of the dielectric permittivities, and μ_p of the magnetic permeabilities of the first $P-1$ dielectric layers. The radius a_P of the PEC core was chosen constant at $k_0a_P = k_0a_{\mathrm{PEC}} = 2\pi$ (one free-space wavelength). In this way, for a medium with P layers, the number of optimization variables for the particles position is $3(P-1)$.

The conducted experiments focused on small values of P in order to obtain designs with a relatively small number of coating layers, which also facilitate the fabrication procedure. For the variations of the variables of the optimization problem, different ranges were considered. Particularly, the differences $k_0(a_{p+1}-a_p)$ between two consecutive layers radii were considered in $\left[\frac{\pi}{10},\pi\right]$ or $\left[\frac{\pi}{10},\frac{\pi}{2}\right]$, while the values of the permittivities ε_p and permeabilities μ_p in [0.5,10], [0.4,5] or [0.5,5].

The external magnetic dipole was taken at $r_0 = 5a_{\mathrm{PEC}}$. The two above-described particle swarm optimization algorithms were developed to minimize the normalized total scattering cross section for a spherical medium with P =3 or 4 total number of layers. The actual reductions in the far-field with respect to the angles of observation are demonstrated in **Figures 4** and **5**, depicting the normalized bistatic scattering cross sections $\sigma(\theta, \phi; \mathbf{r}_0)/\left(\pi a_{\mathrm{PEC}}^2\right)$ versus the angle θ in the xOz and yOz planes, respectively. In these figures, the corresponding cross section curves for a bare (containing no coating layers) PEC sphere are also shown, for comparison purposes.

Significant reductions in the far-field contributions with respect to the bare PEC sphere are observed for large ranges of the observation angles. Particularly, the CAPSO algorithm determines optimal variables corresponding to notably smaller objective function's values for a wide range of observation angles than the classic PSO algorithm. Moreover, the improved performance of the CAPSO algorithm is exhibited by the fact that the attained solutions yield reduced scattered far-field's values for all angles in the yOz plane and for nearly all angles in the xOz plane (apart from a resonance region of the bare PEC cross sections curves around $\theta = 140°$). Another interesting conclusion is that the optimal solutions for $P = 3$ (two covering layers) generate smaller–in general–far field's values for a wider angular range than the optimal solutions for $P = 4$ (three covering layers).

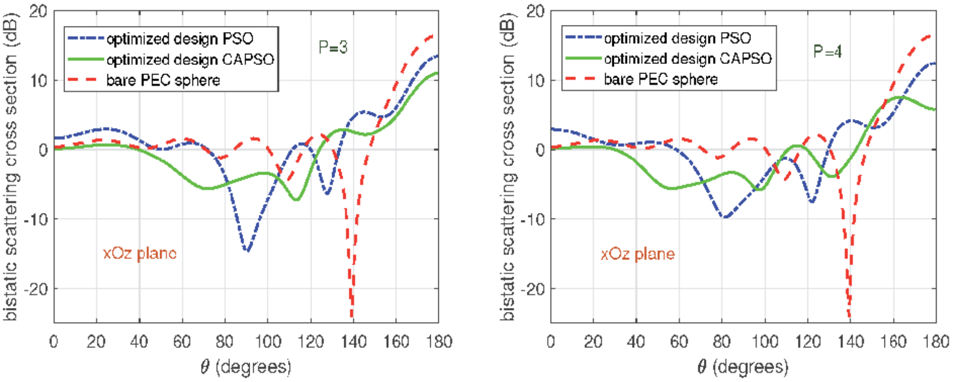


Figure 4.
Normalized bistatic cross section in the xOz plane versus the angle θ for P = 3 (left panel) and P = 4 (right panel) optimized layers with parameters computed by the classic PSO and the CAPSO algorithms.

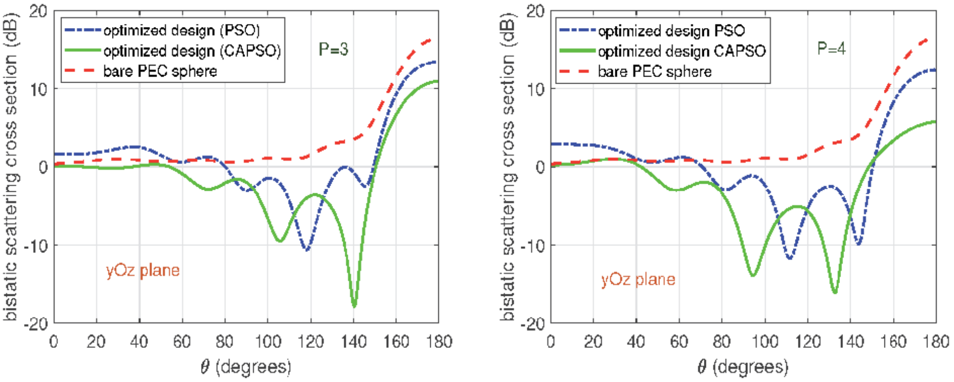


Figure 5.
*As in **Figure 4**, but for the normalized bistatic cross section in the yOz plane.*

Besides, the effectiveness of the cloaking performance of the layered medium with respect to variations of the dipole's distance from the external boundary $r = a_1$ of the medium as well as the sensitivity of the results versus inevitable fabrications imperfections are also important to be examined. Some preliminary numerical results to this direction were presented in [28] by applying the classic PSO algorithm. Extensions to spherical antennas [29] and inhomogeneous media [30] can also be considered by modifying and extending the algorithms presented in this work.

4. Conclusions

Since its introduction to the scientific community, particle swarm optimization (PSO) has gone through many enhancements and variants, and has been applied to numerous diverse problems. The particles that compose the swarm's population act in a manner that follows the basic principles of Swarm Intelligence, as presented in literature. The algorithms utilize the intelligent swarm in order to discover the optima of objective functions. In this chapter, two algorithms were described. The PSO algorithm (1998 version), and the CAPSO algorithm which is a variant of the APSO algorithm. In the PSO, particles move with respect to Newton's laws of motion, and they are described by both position and velocity. Particles' position and velocity updates are affected by the global best $\mathbf{g}^*$ at the time and the individual best $\mathbf{x}^*$. The algorithm includes their learning rates, adjusted in a manner that ensures equal weights to social and individual learning. An inertia mechanism is added to prevent the particles from moving too quickly, thus missing the discovery of optimal solutions. In contrast, the CAPSO algorithm particles do not keep memory of an individual best. They follow a more simplistic approach and update their position in a single step, affected only by the global best at the time. However, there are two parameters, α and β to fine-tune the swarms movement and insert necessary randomness. In CAPSO, the very crucial attraction parameter β, updates through chaotic maps. Specifically, in this work, the Sinusoidal map and the Singer map were considered and applied. It is noted that these maps have a unimode centered around their middle, and have provided the best results in relative research and testing. Both of the discussed algorithms were also provided in pseudocode format.

The PSO and CAPSO algorithms were developed and tested for cloaking problems concerning the covering of a perfectly conducting core by a number of coating layers with optimal parameters so that the total scattered field is significantly reduced. The resulting scattering performance of the medium was examined and it was demonstrated that both PSO and CAPSO algorithms are effective in achieving the goal of the scattered field reduction. Particularly, the CAPSO was shown to be successful in determining optimal solutions yielding enhanced cloaking behavior for a notably large range of the observation angles.

It is noted that the developed algorithms do not utilize a population topology mechanism since the global best is well known to all particles. Thus, in future research, alternative variants of these algorithms could be explored, for example the SPSO 2011 [31] or the Adaptive Clan PSO [32].

Conflict of interest

The authors declare no conflict of interest.

Abbreviations

PSO	Particle Swarm Optimization
APSO	Accelerated Particle Swarm Optimization
CAPSO	Chaotic Accelerated Particle Swarm Optimization
PEC	Perfect Electric Conducting

Author details

Alkmini Michaloglou[†] and Nikolaos L. Tsitsas[*][†]
School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

[*] Address all correspondence to: ntsitsas@csd.auth.gr

[†] These authors contributed equally.

IntechOpen

© 2021 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

References

[1] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks 1995 Nov 27 (Vol. 4, pp. 1942-1948). IEEE.

[2] Heppner F, Grenander U. A stochastic nonlinear model for coordinated bird flocks. The ubiquity of chaos. 1990;233:238.

[3] Shi Y, Eberhart RC. Parameter selection in particle swarm optimization. In: International conference on evolutionary programming 1998 Mar 25 (pp. 591-600). Springer, Berlin, Heidelberg.

[4] Gandomi AH, Yun GJ, Yang XS, Talatahari S. Chaos-enhanced accelerated particle swarm optimization. Communications in Nonlinear Science and Numerical Simulation. 2013 Feb 1;18(2):327-40.

[5] Robinson J, Rahmat-Samii Y. Particle swarm optimization in electromagnetics. IEEE Transactions on Antennas and Propagation. 2004 Apr 5; 52(2):397-407.

[6] Jin N, Rahmat-Samii Y. Particle swarm optimization for antenna designs in engineering electromagnetics. Journal of Artificial evolution and applications. 2008 Mar 30;2008.

[7] Mikki SM, Kishk AA. Quantum particle swarm optimization for electromagnetics. IEEE Transactions on Antennas and Propagation. 2006 Oct 9; 54(10):2764-75.

[8] Mikki SM, Kishk AA. Physical theory for particle swarm optimization. Progress In Electromagnetics Research. 2007;75:171-207.

[9] Goudos SK, Zaharis ZD, Baltzis KB. Particle swarm optimization as applied to electromagnetic design problems. International Journal of Swarm Intelligence Research (IJSIR). 2018 Apr 1;9(2):47-82.

[10] Alù A, Engheta N. Achieving transparency with plasmonic and metamaterial coatings. Physical Review E. 2005 Jul 26;72(1):016623.

[11] Alù A, Engheta N. Multifrequency optical invisibility cloak with layered plasmonic shells. Physical Review Letters. 2008 Mar 18;100(11):113901.

[12] Qiu CW, Hu L, Zhang B, Wu BI, Johnson SG, Joannopoulos JD. Spherical cloaking using nonlinear transformations for improved segmentation into concentric isotropic coatings. Optics Express. 2009 Aug 3;17 (16):13467-78.

[13] Castaldi G, Gallina I, Galdi V, Alù A, Engheta N. Analytical study of spherical cloak/anti-cloak interactions. Wave Motion. 2011 Sep 1;48(6):455-67.

[14] Martins TC, Dmitriev V. Spherical invisibility cloak with minimum number of layers of isotropic materials. Microwave and Optical Technology Letters. 2012 Sep;54(9):2217-20.

[15] Wang X, Chen F, Semouchkina E. Spherical cloaking using multilayer shells of ordinary dielectrics. AIP Advances. 2013 Nov 12;3(11):112111.

[16] Ladutenko K, Peña-Rodríguez O, Melchakova I, Yagupov I, Belov P. Reduction of scattering using thin all-dielectric shells designed by stochastic optimizer. Journal of Applied Physics. 2014 Nov 14;116(18):184508.

[17] Campbell SD, Sell D, Jenkins RP, Whiting EB, Fan JA, Werner DH. Review of numerical optimization techniques for meta-device design.

Optical Materials Express. 2019 Apr 1;9 (4):1842-63.

[18] Wang D, Tan D, Liu L. Particle swarm optimization algorithm: an overview. Soft Computing. 2018 Jan;22 (2):387-408.

[19] Rao SS. Engineering optimization: theory and practice. John Wiley & Sons; 2009 Jul 20.

[20] Millonas MM. Swarms, phase transitions, and collective intelligence. arXiv preprint adap-org/9306002. 1993 Jun 11.

[21] Yang XS. Nature-Inspired Optimization Algorithms. Elsevier; 2014 Feb 17.

[22] Sprott JC. Chaos From Euler Solution of ODEs. Oxford University Press; 2003, pp. 63-65.

[23] Lu H, Wang X, Fei Z, Qiu M. The effects of using chaotic map on improving the performance of multiobjective evolutionary algorithms. Mathematical Problems in Engineering. 2014 Feb;2014.

[24] Tsitsas NL, Athanasiadis C. On the scattering of spherical electromagnetic waves by a layered sphere. The Quarterly Journal of Mechanics and Applied Mathematics. 2006 Feb 1;59(1): 55-74.

[25] Tsitsas NL. Direct and inverse dipole electromagnetic scattering by a piecewise homogeneous sphere. ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik. 2009 Oct 1;89(10):833-49.

[26] Prokopiou P, Tsitsas NL. Electromagnetic excitation of a spherical medium by an arbitrary dipole and related inverse problems. Studies in Applied Mathematics. 2018 May;140 (4):438-64.

[27] Tsitsoglou Z, Prokopiou P, Tsitsas NL. Dipole-Scattering by Spherical Media and Related Optimization Problems. In: 2018 2nd URSI Atlantic Radio Science Meeting (AT-RASC) 2018 May 28 (pp. 1-4). IEEE.

[28] Michaloglou A, Tsitsas NL. Particle Swarm Optimization of Layered Media Cloaking Performance. URSI Radio Science Letters. 2020; 2: (5 pages) DOI: 10.46620/20-0016.

[29] Valagiannopoulos CA, Tsitsas NL. On the resonance and radiation characteristics of multi-layered spherical microstrip antennas. Electromagnetics. 2008 May 27;28(4): 243-64.

[30] Valagiannopoulos CA, Tsitsas NL. Linearization of the T-matrix solution for quasi-homogeneous scatterers. Journal of the Optical Society of America A. 2009 Apr 1;26(4):870-81.

[31] Zambrano-Bigiarini M, Clerc M, Rojas R. Standard particle swarm optimisation 2011 at cec-2013: A baseline for future pso improvements. In: 2013 IEEE Congress on Evolutionary Computation 2013 Jun 20 (pp. 2337-2344). IEEE.

[32] Pontes MR, Neto FB, Bastos-Filho CJ. Adaptive clan particle swarm optimization. In: 2011 IEEE Symposium on Swarm Intelligence 2011 Apr 11 (pp. 1-6). IEEE.

Chapter 6

On the Efficacy of Particle Swarm Optimization for Gateway Placement in LoRaWAN Networks

Clement N. Nyirenda

Abstract

The efficacy of the Particle Swarm Optimization (PSO) in determining the optimal locations for gateways in LoRaWAN networks is investigated. A modified PSO approach, which introduces gateway distancing measures during the initialization phase and flight time, is proposed. For the ease of comparisons and the understanding of the behavior of the algorithms under study, a square LoRaWAN area is used for simulations. Optimization results on a LoRaWAN script, implemented in NS-3, show that the modified PSO converges faster and achieves better results than the traditional PSO, as the number of gateways increases. Results further show that the modified PSO approach achieves similar performance to a deterministic approach, in which gateways are uniformly distributed in the network. This shows that for swarm intelligence techniques such as PSO to be used for gateway placement in LoRaWAN networks, gateway distancing mechanisms must be incorporated in the optimization process. These results further show that these techniques can be easily deployed in geometrically more complex LoRaWAN figures such as rectangular, triangular, circular and trapezoidal shapes. It is generally difficult to figure out a deterministic gateway placement mechanism for such shapes. As part of future work, more realistic LoRaWAN networks will be developed by using real geographical information of an area.

Keywords: Internet of Things, Particle Swarm Optimization, Networks, Simulation, LoRaWAN

1. Introduction

As more and more devices are being embedded with networking capabilities, the Internet of Things (IoT) paradigm is becoming more entrenched in the society. IoT devices communicate with other devices on the Internet seamlessly. To date, IoT has found ample applications in diverse areas such as health, agriculture, safety and security, smart homes, smart water management, smart grids, fleet management and traffic monitoring.

With the accelerated adoption of 5G, companies' plans to invest in IoT solutions will increase even more rapidly. The recent fourth annual Global IoT Executive Survey [1] shows that the number of IoT devices will increase from 8 billion in 2019 to more than 41 billion IoT devices by 2027. Furthermore, the IoT market is geared to grow to over $ 2.4 trillion annually by 2027.

IoT connectivity was primarily based on short range wireless technologies such as Bluetooth mesh networking, Wi-Fi (IEEE 802.11 standard) and Zigbee (IEEE 802.15.4 standard). The trend is, however, shifting toward Low-Power Wide Area Networks (LPWANs), which provide low power, low data rate and long range wireless transmission in the unlicensed frequency bands, configured in a star topology network [2–4]. Interest in LPWANs is further fueled by low deployment costs, large coverage, and the absense of competitors from cellular technologies in the IoT arena [3]. Examples of LPWAN technologies include Sigfox, Narrowband-IoT (NB-IoT) and Long-Range Wide Area Networks (LoRaWAN) [5].

The LoRaWAN technology currently enjoys greater popularity because it is supported by the LoRa Alliance, which is a non-profit association of more than 500 member companies [6]. It is for this reason that this work focuses on this technology. The LoRaWAN network is a star-of-stars topology, where messages are relayed between end devices (EDs) and the central network server (NS) through gateways (GWs). Gateways are linked to the network server through standard IP connections. They act as transparent bridges by converting RF packets to IP packets and vice versa [7].

Although gateways in LoRaWAN networks can cover large areas of end devices, coverage problems still arise when the areas are too big. In such cases, the need to deploy multiple gateways arises [8, 9]. In [8] the impact of redundant packet reception at multiple gateways on data reliability in the LoRaWAN architecture studied, while in [9], an adaptive algorithm for spreading factor selection in LoRaWAN networks with multiple gateways is proposed. In the latter, the maximum number of number of gateways was four and their locations were fixed deterministically.

This work builds on those earlier works by investigating the efficacy of Particle Swarm Optimization (PSO) for gateway placement in a LoRaWAN network. It draws its the motivation from recent studies on PSO based placement of Master Nodes (MNs) in smart water metering networks (SWMNs) [10, 11]. In these networks, Wi-Fi links were used to create a mesh network for the transmission of readings from Smart Meters to Master Nodes. The need to extend PSO approach to LoRaWAN arises naturally because of the advantages of this technology and its growing popularity in the IoT community.

This work adopts a square area, where EDs are deployed randomly and gateways are deployed by using three approaches: 1) the optimal approach based on the standard PSO; 2) the optimal approach based on the PSO algorithm that incorporates gateway distancing mechanisms to prevent gateway interference; 3) and the deterministic approach, where the area is broken down into a number of equally-sized sub-areas, according to the number of gateways and having one gateway deployed at the centre of each sub-area. The LoRaWAN scripts have been implemented using the LoRaWAN NS-3 [12] module [13]. For the PSO approaches, the optimiser which operates at a higher level, invokes the LoRaWAN script, on every function evaluation to calculate the Packet Delivery Ratio for the gateway position configuration created by the algorithm at that moment.

The rest of this Chapter is organized as follows. Section 2 briefly describes the LoRaWAN technology. An overview of Particle Swarm Optimization (PSO) is given in Section 3. Section 4 presents the distancing mechanisms that have been added to standard PSO in order to enhance the spreading of the gateways in the network. Section 5 presents the experimental setup, the results and discussions, and Secton 6 concludes the Chapter.

2. Overview of the LoRaWAN technology

This section gives a brief overview of the aspects of LoRa and LoRaWAN technologies, that are relevant to this study.

2.1 LoRa

LoRa (Long Range) [6] is a physical layer LPWAN technique, developed by Cycleo in France and acquired by Semtech [14]. This technique uses the chirp spread spectrum (CSS) technology, which spreads a narrow-band signal over a wider channel bandwidth. This process greatly enhances the signal's robustness to interference thereby creating long range, low data rate communication over the license-free sub-1GHz Industrial Scientific Medical (ISM) radio bands. The transmission range in LoRa depends on various parameters: bandwidth, coding rate, transmission power, carrier frequency, and six spreading factors (SF), ranging from 7 to 12. These SFs are known to be quasi-orthogonal because they enable simultaneous receptions of packets with different SFs. Another important characteristic is that the increase in SF is accompanied by the signal's resilience to noise, at the expense of data throughput.

2.2 LoRaWAN

LoRaWAN is an upper layer technique that relies on the physical layer LoRa technique. While the LoRa technology is proprietary, LoRaWAN is an open standard, developed and supported by the LoRa Alliance [6]. It is an ALOHA-based protocol which organizes networks in a star-of-stars topology, with the following major components:

1. *End devices (EDs)*, which generate uplink data and send it to the network server through the gateway through a single-hop LoRa communication. EDs also receive downlink traffic from the gateways.

2. *Gateways (GW)*, which serve as link between the EDs and the network servers by using and IP backbone. They collect data sent by EDs and forward it to network servers. They also relay packets sent by the network servers to the EDs.

3. *Network Server (NS)*, which serves as the central coordinator and controller of the LoRaWAN network.

The LoRaWAN standard defines three types of EDs, namely Class A, Class B, and Class C. Class A is the default class which must be supported by all LoRaWAN EDs. In this class, communication is always initiated by the ED and is fully asynchronous. Uplink transmission is followed by two short downlink windows, to cater for bi-directional communication and/or the transmission of network control commands. In addition to all the components of Class A, Class B EDs provide regular receive windows for potential downlink traffic. In Class C, the EDs remain in continuous receive mode thereby reducing latency on the downlink path.

The LoRaWAN standard also supports an Adaptive Data Rate (ADR) scheme, which enables the NS to maximize both battery battery life of the EDs and overall network capacity, by setting the data rate (DR) and RF output power for each ED individually. When radio conditions are bad, data rate is lowered by increasing th SF, leading to low coverage. Conversely, when radio conditions are good, data rate is increased by lowering the SF or by reducing the transmit power of a node in order to maximize the battery lifetime and optimize overall network capacity. LoRaWAN also suffers from collisions of packets at the gateways [9, 15–17]. This happens when two or more signals arrive at the the gateway in same time duration. These collisions lower the data rates drastically.

Furthermore, this standard also faces the problem of network coverage when the network area is very large. In this case, one GW cannot cater for all the EDs in the area. As a result, there is a need for multiple gateways. To ensure that the network is optimally covered, there is a need for sufficient inter-gateway distance. The work in [9] implemented a maximum of 4 gateways in a 16 km × 16 km square area, in which the gateways are located at the centres of the four quadrants. The results of this work show that higher packet delivery rates are achieved with 4 GWs as the network coverage area increases. In [8], the impact of redundant packet reception at multiple gateways on data reliability is studied. This work models the Average Successful Transmission Probability (ASTP) as a function of end device density, gateway density, and traffic intensity thereby providing useful insights into the deployment of multiple gateways in LoRaWANs.

Since this work focuses on the determining the effectiveness of Particle Swarm Optimization (PSO) for the placement of multiple gateways, the next section presents an overview of the standard PSO algorithm.

3. The standard particle swarm optimization algorithm

Introduced by Kennedy and Eberhart in 1995, the Particle Swarm Optimization (PSO) draws inspiration from the social behavior of animals living in swarms, such as flocks of birds [18]. PSO is initialized with a population of N particles that are generated randomly in a pre-determined search space S of D dimensions. Each particle denotes a candidate solution to a problem and is characterized by three main parameters: its *current position*, *current velocity* and the *best position* ever found by the particle during the search process. The particles fly in the search space in order to find the optimal solution. The trajectory of a particle is influenced by the particle's own experience as well as it's neighboring particles. The velocity of the i-th particle is updated at every iteration by using

$$v_i(t) = \omega * v_i(t-1) + c_1 r_1 \big(p_i^b - x_i(t)\big) + c_2 r_2 \big(g^b - x_i(t)\big), \tag{1}$$

where $i = 1, 2, \ldots, N$; c_1 and c_2 are constants denoting cognitive and social parameters respectively.

The values of c_1 and c_2 are chosen in the range $[0.5, 2.5]$. They are applied to cater for the influence of the particle's historical best position p_i^b and the swarm's best position g^b respectively. Parameters r_1 and r_2 are random numbers uniformly distributed within $[0, 1]$, while ω denotes the inertia weight; it helps to dampen the velocities of the particles to assist in the convergence to the optimum point at the end of the optimization iteration.

In order to keep the particle's velocity bounded, a further arbitrary parameter $V_m = (v_{m1}, v_{m2} \ldots v_{mD}) \in S$, was defined. Whenever, a vector element, exceeds the corresponding element of V_m, the element is reset to its upper limit. Once the velocity has been updated by using Eq. (1), the position of each particle is updated at each iteration by using

$$x_i(t+1) = x_i(t) + v_i(t+1). \tag{2}$$

In terms of implementation, the PSO algorithm goes through the following steps:

1. *Initialization*: It begins by initiliazing N, c_1, c_2, ω, and G, which denotes the maximum number of iterations. Then the initial population of N particles is generated with random values from the search space and initial velocities for each particle are set to 0. The fitness function values for all N particles are evaluated based on their initial positions. The initial positions of each particle are set as the personal best positions for the respective particles, and the overall best position found so far is set as best solution for the swarm.

2. *Flight time*: Once the initialization process is done, the algorithm goes into the iterative process. At each iteration, the particle positions in the search space are updated using Eqs. (1) and (2); fitness function values of all particles are updated based on their new positions. If necessary, personal best and global best values are updated accordingly.

3. *Termination*: This iterative process is terminated once G iterations are completed. At this point, the best solution for the swarm becomes the optimal solution for the optimization run.

The next section discusses the gateway distancing measures that have been incorporated in the standard PSO algorithm in order to spread out the locations of gateways in the LoRaWAN network.

4. Gateway distancing measures in PSO

This section presents the gateway distancing measures that were incorporated in the PSO algorithm. Properly distanced GWs will help to ensure that the network coverage is high, thereby increasing the packet delivery rate (PDR). Before presenting these measures, the GW placement optimization problem has to be presented.

4.1 The GW placement optimization problem

The GW placement optimization problem adopts the approach used in the Master Node optimization problem in [10, 11]. A LoRaWAN network is assummed to contain n_{ed} EDs in a rectangular area defined by $L \times M$, where L and M are in kilometers. The number of GWs in the network is denoted by n_{gw}. The location of each GW is defined by the x and y coordinates. As a result, the number of variable parameters in the vector of GW coordinates that depicts the locations of all the n_{gw} GWs is $2 * n_{gw}$. In PSO terminology, this vector is called a particle. For instance, for a LoRaWAN network with 4 GWs, the number of parameters in the particle will be 8. The aim of the optimization process is to obtain the particle that achieves the highest packet delivery ratio (PDR), where PDR is defined by

$$PDR = P_r/P_s, \tag{3}$$

where P_r is the number of packets received at the NS, excluding duplicate packets, while P_s denotes the number of packets sent by by the EDs. The location information for the $2 * n_{gw}$ GWs can be encoded in a particle by using

$$\mathbf{p} = (p_0, p_1 \cdots p_{D-1}), \tag{4}$$

where p_0 and p_1 are the coordinates of the first GW; p_2 and p_3 are the coordinates of the second GW; p_D and p_{D-1} are the coordinates of the final GW; $D = 2 * n_{gw}$. If the standard PSO is used, each even-indexed element of the particle is defined within $[0, L]$, while each odd-indexed element is defined within $[0, M]$, denoting the x and y-cordinates respectively. When GW distancing measures are employed, this process is modified as explained in the next subsection.

4.2 The PSO algorithm with GW distancing measures

GW distancing measures can be applied during the initialization process as well as during flight time. During initialization, the initial GW positions can be set in such a way that they are sufficiently far away from each other. During the iterative process, only those particles that depict sufficient average inter-gateway distance are evaluated. Next, the two techniques that would address GW distancing requirements will be presented.

4.2.1 GW distancing during initialization

This technique aims at initiliazing the locatons for the respective GWs to some equal but distinct sub-areas of the LoRaWAN. The lengths of the sides of each of those sub-areas be denoted by ΔL and ΔM can be defined by

$$\Delta L = L/\sqrt{n_{gw}} \tag{5}$$

and

$$\Delta M = M/\sqrt{n_{gw}}. \tag{6}$$

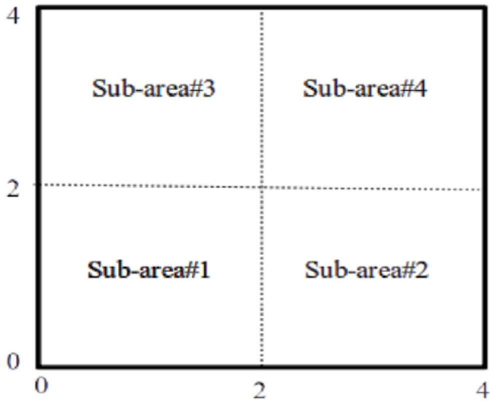


Figure 1.
Illustration of the sub-areas used in the GW initialization process.

For instance, if 4 GWs are initialized in a square LoRaWAN area of sides 4 km by 4 km, $L = M = 4km$, ΔL and ΔM would both be equal to 2 km as illustrated in **Figure 1**. Each GW will be initialized in the distinct sub-area such that the coordinates for the four GWs will be initialized as follows: for GW1, $x \in [0, 2]$ and $y \in [0, 2]$; for GW2, $x \in [0, 2]$ and $y \in [2, 4]$; for GW3, $x \in [0, 2]$ and $y \in [2, 4]$; and for GW4, $x \in [2, 4]$ and $y \in [2, 4]$.

For some LoRaWAN network areas and designated number of GWs, it will not be possible to fit all the GWs into the network by using the aforementioned approach. In such cases, the extra GWs can be randomly initialized with coordinates drawn from the entire area. The number of extra GWs n_{egw} can be determined by using

$$n_{egw} = n_{gw} - (M/\Delta M * L/\Delta L) \tag{7}$$

4.2.2 GW distancing during flight time

The parameter that governs GW distancing during flight time is the average inter-gateway distance d_k, which is defined for each particle k in the swarm by using

$$d_k = \frac{1}{n_e} \sum_{i=0}^{n_{gw}} \sum_{j=i+1}^{n_{gw}} \sqrt{\left(p_{k,2i} - p_{k,2j}\right)^2 + \left(p_{k,2i+1} - p_{k,2j+1}\right)^2}, \tag{8}$$

where n_e is the number of edges among the GWs, which is defined by

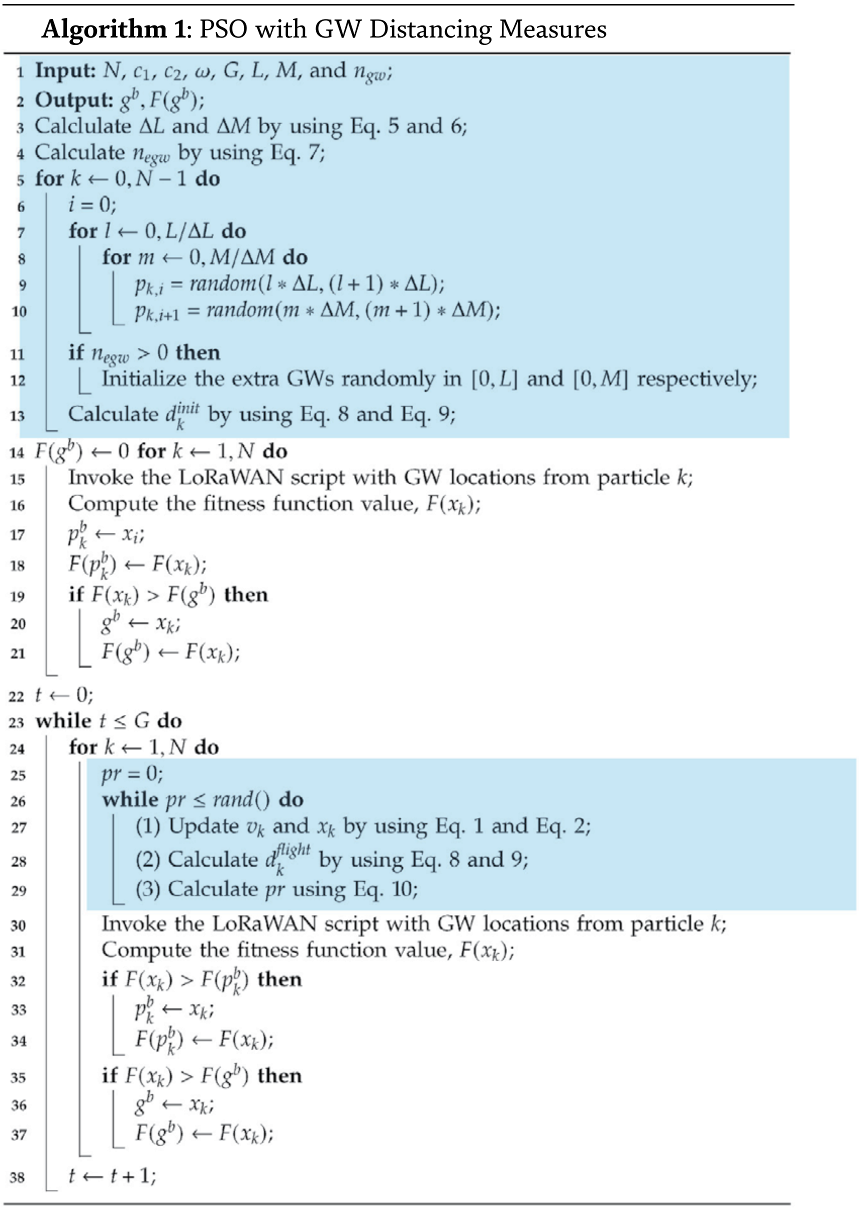
$$n_e = \frac{n_{gw}\left(n_{gw} - 1\right)}{2}. \tag{9}$$

After the initialization process, the initial average inter-gateway distance d_k^{init} is calculated for all particles by using Eqs. (8) and (9). During flight time, an inter-gateway distance d_k^{flight} is calculated after every particle position update using the same equations. Fitness function evaluation for the particle k is triggered by using the probabilty pr which is defined by

$$pr = \begin{cases} \dfrac{d_k^{flight}}{d_k^{init}}, & \text{if } d_k^{flight} \le d_k^{init} \\ 1, & \text{otherwise} \end{cases} \tag{10}$$

The decision on whether to evaluate particle k or not is governed by the following rule: if $rand() < pr$, then evaluate particle k, else try to get new position for the particle and test the rule again. The parameter $rand()$ is a random number in the range $[0.0, 1.0]$. This process will continue until the rule fires.

Algorithm 1 shows the modified PSO algorithm, with the GW distancing measures highlighted in blue color. On lines 15 and 30, the LoRaWAN script is invoked and the PDR value emanating from that process is construed as the fitness function value, $F(x_k)$, for each particle k.



Algorithm 1: PSO with GW Distancing Measures

```
Input: $N, c_1, c_2, \omega, G, L, M$, and $n_{gw}$;
Output: $g^b, F(g^b)$;
Calclulate $\Delta L$ and $\Delta M$ by using Eq. 5 and 6;
Calculate $n_{egw}$ by using Eq. 7;
for $k \leftarrow 0, N-1$ do
    $i = 0$;
    for $l \leftarrow 0, L/\Delta L$ do
        for $m \leftarrow 0, M/\Delta M$ do
            $p_{k,i} = random(l * \Delta L, (l+1) * \Delta L)$;
            $p_{k,i+1} = random(m * \Delta M, (m+1) * \Delta M)$;
    if $n_{egw} > 0$ then
        Initialize the extra GWs randomly in $[0, L]$ and $[0, M]$ respectively;
    Calculate $d_k^{init}$ by using Eq. 8 and Eq. 9;
$F(g^b) \leftarrow 0$ for $k \leftarrow 1, N$ do
    Invoke the LoRaWAN script with GW locations from particle $k$;
    Compute the fitness function value, $F(x_k)$;
    $p_k^b \leftarrow x_i$;
    $F(p_k^b) \leftarrow F(x_k)$;
    if $F(x_k) > F(g^b)$ then
        $g^b \leftarrow x_k$;
        $F(g^b) \leftarrow F(x_k)$;
$t \leftarrow 0$;
while $t \leq G$ do
    for $k \leftarrow 1, N$ do
        $pr = 0$;
        while $pr \leq rand()$ do
            (1) Update $v_k$ and $x_k$ by using Eq. 1 and Eq. 2;
            (2) Calculate $d_k^{flight}$ by using Eq. 8 and 9;
            (3) Calculate $pr$ using Eq. 10;
        Invoke the LoRaWAN script with GW locations from particle $k$;
        Compute the fitness function value, $F(x_k)$;
        if $F(x_k) > F(p_k^b)$ then
            $p_k^b \leftarrow x_k$;
            $F(p_k^b) \leftarrow F(x_k)$;
        if $F(x_k) > F(g^b)$ then
            $g^b \leftarrow x_k$;
            $F(g^b) \leftarrow F(x_k)$;
    $t \leftarrow t + 1$;
```

5. Simulation environment, results and discussions

Since the goal of this work is to determine the efficacy of PSO in GW placement in LoRaWAN networks, the standard PSO and the modified PSO (with distancing measures) are compared with a deterministic approach. For brevity, the standard PSO, will be referred to as PSO, while the modified PSO will be refered to as PSODIST. For purposes of ease of comparison and discussion of the ensuing results, a square LoRaWAN area is adopted. Parameters L and M are both set to 50 km and 50 km respectively. The number of GWs n_{gw} is varied by using 4, 9, 25, and 49, which are all square numbers. This caters for the ease of deployment in the deterministic approach, which uniformly distributes the GWs as illustrated in **Figure 1** for 4 GWs. The expectation is that such a deterministic approach will yield the best result for the square LoRaWAN network area. If a PSO based approach can achieve similar results to the deterministic approach, the effectiveness of the PSO approach for GW placement will be proved.

5.1 The LoRaWAN script and simulation parameters

The LoRaWAN script is implemented in the NS-3 LoRaWAN module [13]. In order to reduce the simulation time for the LoRaWAN script during the optimization process, the number of EDs, n_{ed} is set 400. These EDs are set up randomly in the LoRaWAN area and the resulting text file is saved on the system. The ED locations are loaded to the simulation together with the GW locations on every LoRaWAN script invocation.

With the introduction of multiple GWs, a single packet may find its way to the Network Server (NS) through two or more GWs. In order to remove such packets from the number of unique packets received at the NS, some modifications were made to the LoRaWAN module. The default LoRa physical layer parameters in [13] were used in the simulations. The rest of the parameters in the LoRaWAN script are shown in **Table 1**. For the PSO algorithms, parameters are set as follows:$N = 20$, $c_1 = 1.5$, $c_2 = 1.5$, $\omega = 0.7$, and $G = 50$. They are all within the ranges that are commonly used in literature.

The basic C++ PSO code used in this study have been downloaded from [19]. The PSODIST code was developed by incorporating the GW distancing measures, explained in Section 4, in the basic PSO code. A personal computer with an Intel® Core™ i7-7500U CPU @ 2.70GHz × 4 processor with 8 GB RAM, running on Ubuntu 18.04.5 LTS, was used in this study. Ten optimization runs were conducted for each specific number of GWs for the PSO and PSODIST approaches. The LoRaWAN script was seeded with the same values, in order to ensure that the same simulation conditions prevail in all the optimization runs.

5.2 The efficacy of GW distancing measures

In this subsection, the efficacy of the GW distancing measures in the PSODIST algorithm is investigated by examining the mechanics of PSODIST against the basic PSO approach. **Figures 2–5** show the evolution of the optimization process for PSO and PSODIST approaches for the best optimization runs for 4, 9, 25, and 49 GWs. In the case of 4 GWs, in **Figure 2**, the difference in the evolution of the PDR value between the two approaches is very minimal. In fact, contrary to expectation, the PSODIST approach starts from a worse off position at around 36% while PSO starts from 38%. The GW distancing process during the initialization process does not help simply because the partitions in which the GWs are initialized are too big. The flight time GW distancing process helps to push PSODIST barely above the PSO. It can, therefore, be observed that GW distancing measures do not improve the performance of PSO when the number of GWs is low.

Parameter	Value
Spreading factors	7, 8, 9, 10, 11 and 12
GW height	20 m
ED height	2 m
Packet generation rate	12 packets/h
Simulation time	3000 s

Table 1.
LoRaWAN simulation parameters.

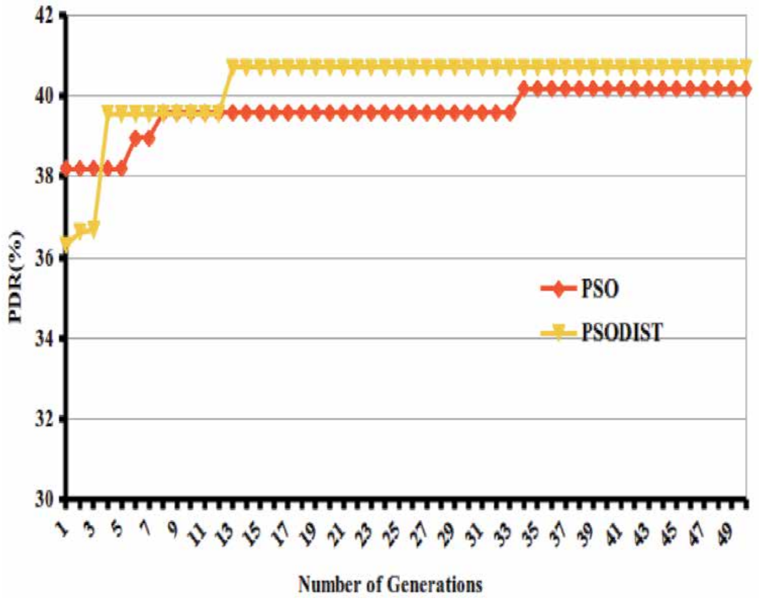


Figure 2.
PDR values of the best particle in the PSO swarms as the number of generations increases, with the number of GWs set to 4.

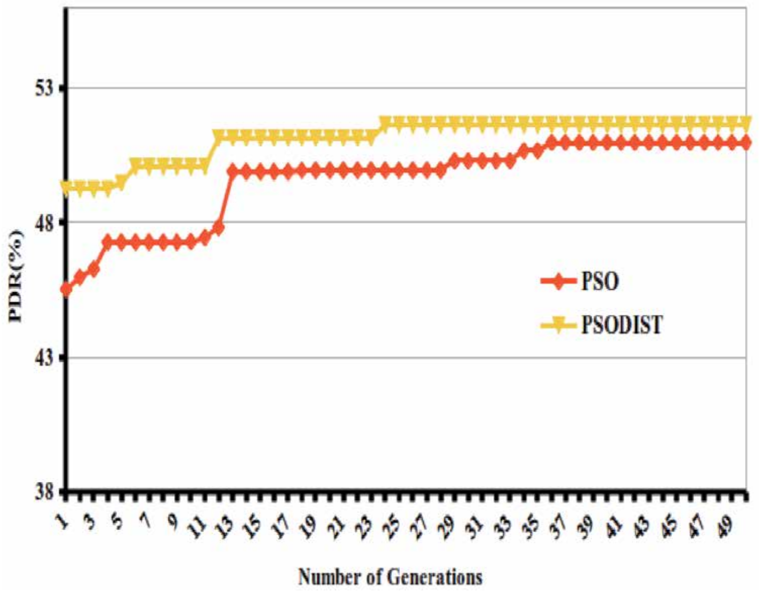


Figure 3.
PDR values of the best particle in the PSO swarms as the number of generations increases, with the number of GWs set to 9.

In the case of 9 GWs, as shown in **Figure 3**, PSODIST starts with a PDR of 49.2% while PSO starts with 45.5%. The trend becomes even more prominent as the number of GWs increases (see **Figures 4** and **5**). At 25 GWs, PSO and PSODIST register initial PDR values of 47.4% and 52.3% respectively, while at 49 GWs, the initial PDR values are 47% and 49.9% respectively. This shows that the benefits of GW distancing measures in the initializition process are only realized as the number

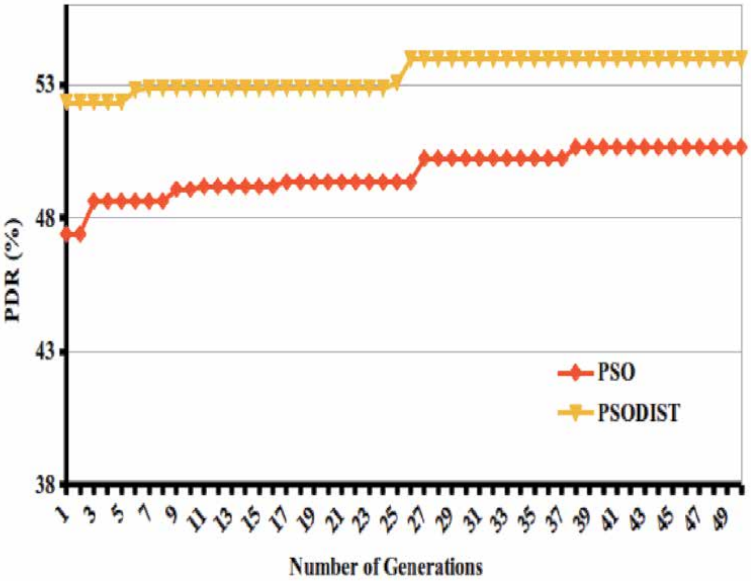


Figure 4.
PDR values of the best particle in the PSO swarms as the number of generations increases, with the number of GWs set to 25.

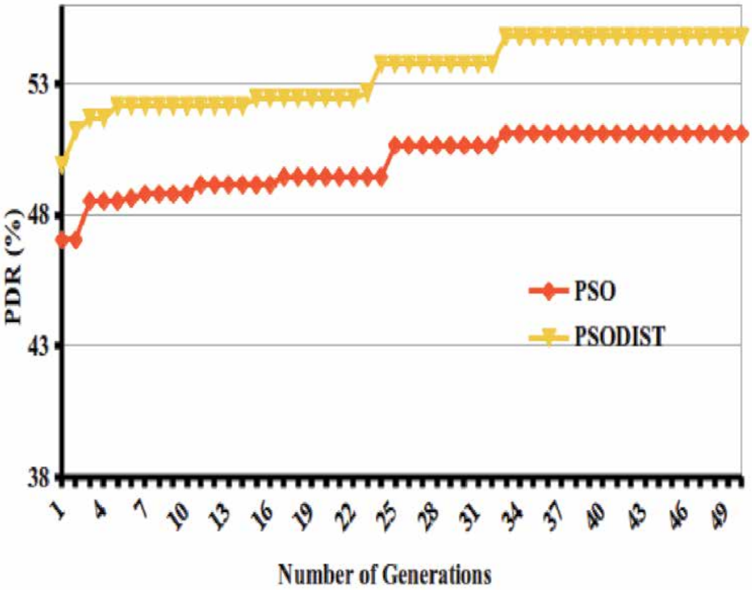


Figure 5.
PDR values of the best particle in the PSO swarms as the number of generations increases, with the number of GWs set to 40.

of GWs increases. This is due to the reduction in the sizes of the partitions in which the GWs are initialized.

As the number of GWs increases, the flight time GW distancing process seems to be more effective than in the case of 4 GWs. **Figure 3** shows that at 9 GWs, PSODIST converges to a PDR value of 51.63% at the 24th generation, while PSO converges to 50.98% at the 38th generation. In the case of 25 GWs, as shown in **Figure 4**, PSODIST converges to a PDR value of 53.95% at the 27th generation, while PSO converges to 50.65% at the 38th generation. When 49 GWs are used, as shown in **Figure 5**, PSODIST converges to a PDR value of 54.8% at the 34th generation, while PSO converges to 51.1% at the 37th generation. These results show that the basic PSO suffers from delayed convergence as well as suboptimal convergence because the initial positions of the GWs are not properly distanced. It can, therefore, be concluded that GW distancing measures improve the performance of PSO when the number of GWs is increased.

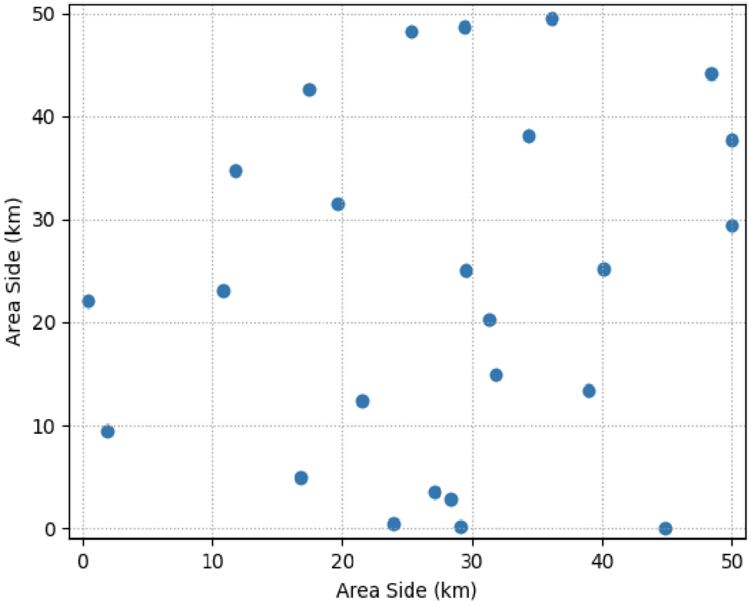


Figure 6.
Locations of 25 GWs in the 50 km × 50 km area, as depicted by the best particle in the PSO swarm.

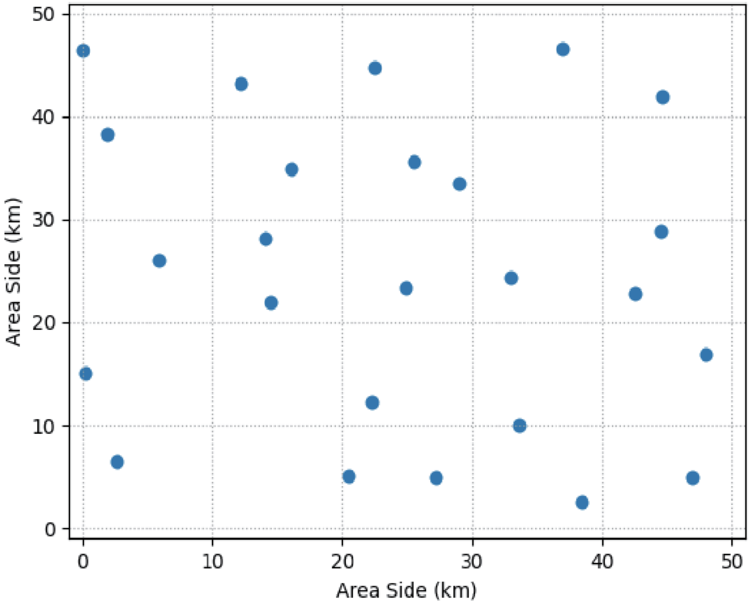


Figure 7.
Locations of 25 GWs in the 50 km × 50 km area, as depicted by the best particle in the PSODIST swarm.

Figures 6–9 illustrate the spread of the GW locations in the LoRaWAN area. From these graphical presentations, it is easy to see that the PSODIST achieves a better GW spread than PSO. In the PSO approach, there are some sections where the GWs are too crowded. For instance, in **Figure 6**, the subarea bounded by coordinates (20, 0), (20, 10, 30, 10, 30, 0), has four GWs, while in the north-western corner, there are no GWs in a subarea that is 2.5 times the former. In **Figure 8**, a similar trend is observed as there are even up 6 GWs in the central

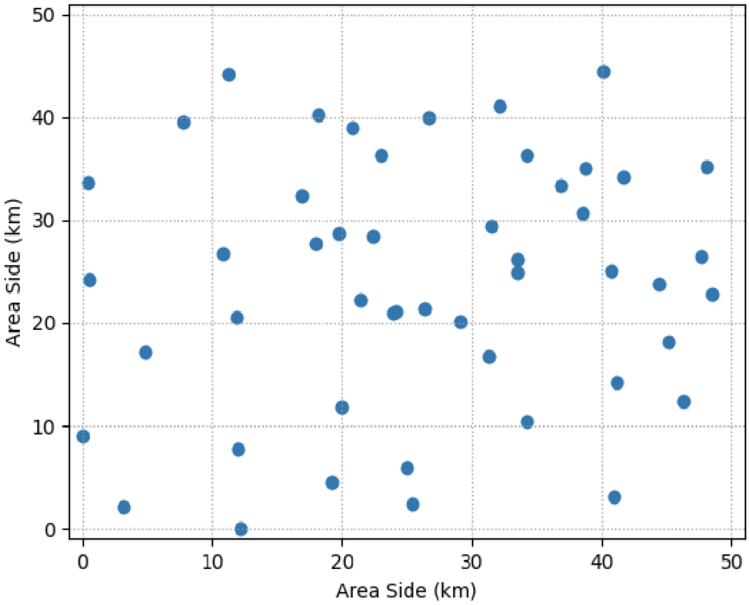


Figure 8.
Locations of 49 GWs in the 50 km × 50 km area, as depicted by the best particle in the PSO swarm.

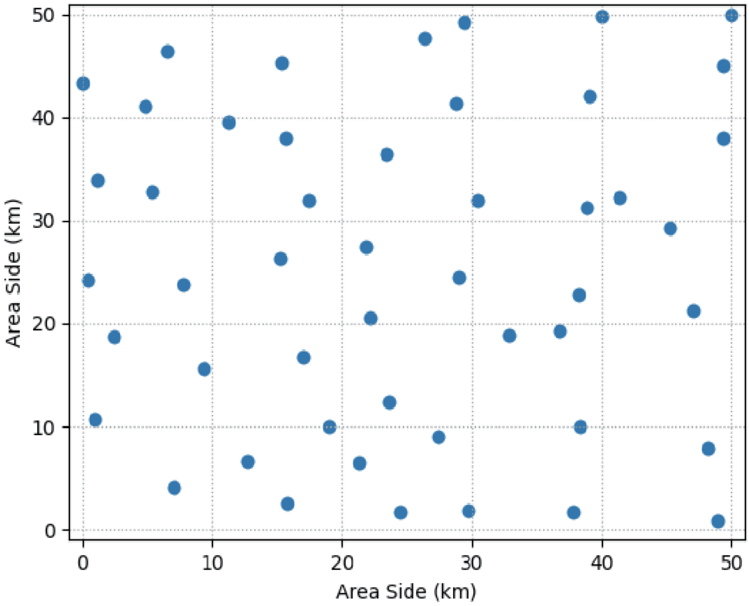


Figure 9.
Locations of 49 GWs in the 50 km × 50 km area, as depicted by the best particle in the PSODIST swarm.

10 km x 10 km subarea. The EDs in the subareas with little GW coverage suffer from GW reachability issues. On the other hand, subareas with many GWs also suffer from increased downlink traffic interference. Both of these conditions lead to the reduction of PDR.

5.3 Benchmarking PDR performance of PSO with the deterministic approach

A separate simulation exercise was conducted using the deterministic approach for benchmarking purposes. Ten simulation runs were also conducted for 4, 9, 25, and 49 GWs. **Table 2** shows the minimum, mean and maximum PDR values from the two PSO approaches along with the deterministic approach. Results depicting the best performing approach are shown in bold text. For 4 GWs, there is no significant difference among the three approaches as best minimum value of 39.8% is from PSO. On the other hand, the deterministic approach and PSODIST achieve the best mean and maximum values of 40.23% and 40.70% respectively. With 9 GWs, PSODIST seems to have an upper hand in the sense that it achieves the best minimum and mean values of 51.28% and 51.51% respectively, while the deterministic approach gets the best maximum value of 52.38%. At 25 GWs, all the best

No. of GWs	Deterministic			PSO			PSODIST		
	Min	**Mean**	**Max**	**Min**	**Mean**	**Max**	**Min**	**Mean**	**Max**
4	38.66	**40.23**	42.68	**39.58**	40.06	40.58	39.55	40.17	**40.70**
9	48.15	49.95	**52.38**	48.28	49.81	50.98	**51.28**	**51.51**	51.63
25	49.70	51.78	53.28	50.65	51.50	52.15	**53.50**	**53.74**	**53.95**
49	**54.25**	**55.29**	**57.18**	50.43	50.81	51.10	53.50	54.33	54.80

Table 2.
Benchmarking of PSO approaches with the deterministic approach.

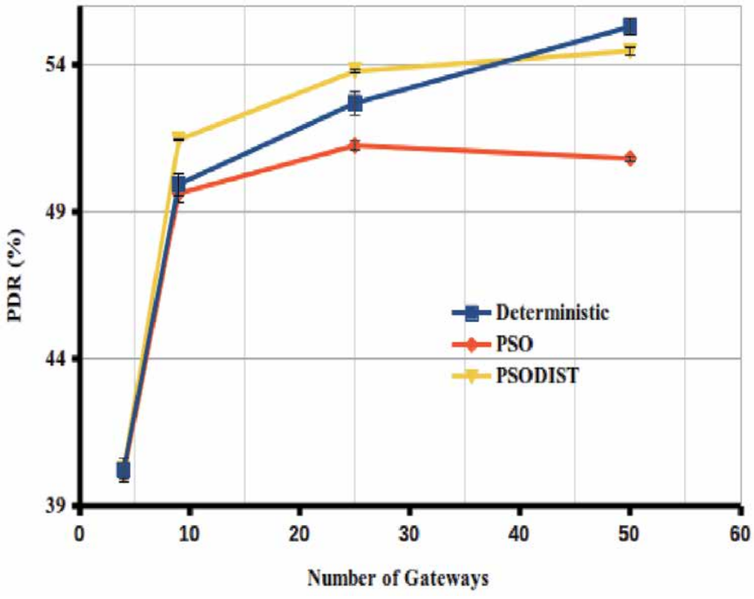


Figure 10.
A comparison of the PDR values for the three approaches when the number of GWs is set to 4, 9, 25 and 50.

values come from PSODIST, while at 50GWs, they all come from the deterministic approach. The basic PSO seems not to compete well with the other two approaches.

Figure 10 illustrates the same results from a graphical perspective, where the performance of PSODIST compares fairly well with the deterministic approach. It even outperforms the deterministic approach at 25 GWs. This clearly shows that when PSO approach is used for GW placement in LoRaWAN networks, there is a need for GW distancing mechanisms in order to prevent GWs from accumulating in some specific zones thereby jeopardizing performance in terms of PDR.

6. Conclusions

This Chapter investigated the efficacy of using Particle Swarm Optimization (PSO) in determining the optimal locations for gateways in large LoRaWAN networks. The coordinates of the gateways are coded into a vector, which denotes particle in PSO terminology. Gateway distancing measures have been proposed for the initiliazation and flight time phases of the PSO in order to spread out the gateways in the network. This process created a modified PSO algorithm, herein refered to as PSODIST. During initialization, the LoRaWAN area is broken down into a number of subareas and each gateway is initiliazed in a specific area. If there are some extra gateways, which cannot fit in the subareas, such gateways are initialized randomly within the entire area. During flight time, only new particle positions, that exhibit a sufficiently high inter-gateway distance, are evaluated. Optimization experiments are conducted to verify the effectiveness of the PSODIST approach. The LoRaWAN script, used in the optimization process, is implemented in NS-3 [12] using the recently proposed LoRaWAN module [13]. The function evaluation routines in the PSO algorithms invoke the LoRaWAN script and the resulting packet delivery rate (PDR) is retained as the fitness for the respective particle.

The mechanics of the optimization process show that there is no difference between PSODIST and PSO when the number of gateways is small. Nevertheless, as the number of gateways increases, the impact of distancing measures becomes evident; PSODIST yields better optimization rates as well as much faster convergence than PSO. The much more even spread of the gateway locations determined by PSODIST has also been demonstrated graphically. The results from the PSO approaches have been further compared with the deterministic approach which arranges the gateways uniformly over the LoRaWAN area. PSODIST yields similar PDR values as the deterministic approach in the 50 km × 50 km LoRaWAN area. Unlike the deterministic approach, which relies on square numbers and square areas for uniform coverage, PSODIST can work well for any number of GWs. It is, therefore, possible to get optimal performance at a lower number of GWs, in between square numbers. Furthermore, PSODIST is not shape-dependent. It can, therefore, be deployed easily in geometrically more complex LoRaWAN figures such as rectangular, triangular, circular and trapezoidal shapes. The development of more realistic LoRaWAN network by using real geographical information of an area will be considered in future.

Abbreviations

IoT	Internet of Things
NB	Narrowband
LoRaWAN	Long Range Wide Area Network

LPWAN	Low-Power Wide Area Networks
LoRa	Long Range
PSO	Particle Swarm Optimization
ED	End Devices
GW	Gateway
GW	Network Server
SWMN	Smart Water Metering Network
SWMN	Chirp Spread Spectrum
SF	Spreading Factor
PDR	Packet Delivery Ratio

Author details

Clement N. Nyirenda[†]
Faculty of Natural Sciences, Department of Computer Science, University of the Western Cape, Cape Town, South Africa

[*]Address all correspondence to: cnyirenda@uwc.ac.za

† These authors contributed equally.

IntechOpen

© 2021 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

References

[1] Preview of the Business Insider Intelligence The Internet of Things 2020 premium research report. https://www.businessinsider.com/internet-of-things-report?IR=T [Accessed: 2021-01-27]

[2] Magrin D, Centenaro M, Vangelista L. Performance evaluation of LoRa networks in a smart city scenario. In: Proceedings of the IEEE International Conference on Communications (ICC '17); May 2017, Paris, France. New York: IEEE; 2017. p. 1-7.

[3] Centenaro M, Vangelista L, Zanella A, and Zorzi M, "Long-Range Communications in Unlicensed Bands: the Rising Stars in the IoT and Smart City Scenarios," IEEE Wireless Commun.,2016;23(5):60–67

[4] Kufakunesu R, Hancke GP, Abu-Mahfouz AM. A survey on adaptive data rate optimization in LoRaWAN: Recent solutions and major challenges. Sensors. 2020 Jan;20(18):5044.

[5] Mekki K, Bajic E, Chaxel F, Meyer F. A comparative study of LPWAN technologies for large-scale IoT deployment. ICT express. 2019 Mar 1;5 (1):1-7.

[6] LoRA Alliance. https://lora-alliance.org/ [Accessed: 2021-01-27]

[7] LoRAWAN Network. https://lora-alliance.org/about-lorawan/ [Accessed: 2021-01-27]

[8] Ni M, Jafarizadeh M, Zheng R. On the effect of multi-packet reception on redundant gateways in lorawans. In: Proceedings of the IEEE International Conference on Communications (ICC 2019); May 2019, Shanghai, China. New York: IEEE; 2019. p. 1-6.

[9] Loubany A, Lahoud S, El Chall R. Adaptive algorithm for spreading factor selection in LoRaWAN networks with multiple gateways. Computer Networks. 2020;182:107491.

[10] Nyirenda CN, Nyirongo SG. Binary Search Based PSO for Master Node Enumeration and Placement in a Smart Water Metering Network. In: Proceedings of the International Conference on e-Infrastructure and e-Services for Developing Countries. Dec 3 2019; Springer, Cham; p. 104-118.

[11] Nyirenda CN, Makwara P, Shitumbapo L. Particle swarm optimization based placement of data acquisition points in a smart water metering network. In: Proceedings of SAI Intelligent Systems Conference. September 21, 2016; Springer, Cham; p. 905-916.

[12] NS-3 Simulator.https://www.nsnam.org/ [Accessed: 2021-01-27]

[13] LoRaWAN ns-3 module https://github.com/signetlabdei/lorawan [Accessed: 2021-01-27]

[14] Semtech Acquires Wireless Long Range IP Provider Cycleo https://www.design-reuse.com/news/28706/semtech-cycleo-acquisition.html [Accessed: 2021-01-27]

[15] Markkula J, Mikhaylov K, Haapola J. Simulating LoRaWAN: On importance of inter spreading factor interference and collision effect. In: Proceedings of the 2019 IEEE International Conference on Communications (ICC '19); May 20, 2019; p. 1-7; IEEE.

[16] Marini R, Cerroni W, Buratti C. A novel collision-aware adaptive data rate algorithm for LoRaWAN networks. IEEE Internet of Things Journal. DOI: 10.1109/JIOT.2020.3020189

[17] Triantafyllou A, Sarigiannidis P, Lagkas T, Sarigiannidis A. A Novel

LoRaWAN Scheduling Scheme for Improving Reliability and Collision Avoidance. In2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST); Sep 7, 2020; p. 1-4; IEEE.

[18] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks. Nov 27, 1995 Nov 27; Vol. 4, IEEE; p. 1942-1948.

[19] PSO code http://profesores.elo.utfsm.cl/∼tarredondo/code/simgalib/1-7-1/ [Accessed: 2021-01-27]

Chapter 7

Pareto-Based Multiobjective Particle Swarm Optimization: Examples in Geophysical Modeling

Ersin Büyük

Abstract

It has been recently revealed that particle swarm optimization (PSO) is a modern global optimization method and it has been used in many real world engineering problems to estimate model parameters. PSO has also led as tremendous alternative method to conventional geophysical modeling techniques which suffer from dependence to initial model, linearization problems and being trapped at a local minimum. An area neglected in using PSO is joint modeling of geophysical data sets having different sensivities, whereas this kind of modeling with multiobjective optimization techniques has become an important issue to increase the uniqueness of the model parameters. However, using of subjective and unpredictable weighting to objective functions may cause a misleading solution in multiobjective optimization. Multiobjective PSO (MOPSO) with Pareto approach allows obtaining set of solutions including a joint optimal solution without weighting requirements. This chapter begins with an overview of PSO and Pareto-based MOPSO presented their mathematical formulation, algorithms and alternate approaches used in these methods. The chapter goes on to present a series synthetic modeled of seismological data that is one kind of geophysical data by using of Pareto-based multiobjective PSO. According to results matched perfectly, we believe that multiobjective PSO is an innovative approach to joint modeling of such data.

Keywords: particle swarm optimization, Pareto-based optimization, multiobjective, geophysical modeling

1. Introduction

As a conventional approach, least squares and linear programming optimization methods have been used modeling of geophysical data with a general form without requiring any special case. However, due to some disadvantages of these methods such as computational time and linearization problems, it has become inevitable to tend to new approaches for the researchers. Unlike traditional optimization methods, optimization of nonlinear models has been improved in two ways which are derivative based and non-derivative search methods. Unfortunately, one of the major disadvantages of derivative based methods is that solutions potentially trap at a local minimum because of depending on initial model. On the other hand, non-derivative search methods partly provide a global solution, however, significantly increase computing time.

In recent years, two approaches that are artificial intelligence and meta-heuristic optimization algorithms have been effectively put forward in geophysical modeling studies having complex and nonlinear models. Meta-heuristic optimization algorithms called as modern global optimization methods are based on systematic characteristics of biological, molecular, neurobiological and animal swarms [1]. PSO as one of the modern global optimization algorithms, has increased its popularity with rapid convergence compared to various optimization algorithms [2, 3], especially when real model parameters are used [4].

PSO for multiobjective optimization has also been used in many studies in order to solve real world engineering problems having conflicted solutions between objective functions [5]. Despite this interest, very few researchers have studied MOPSO for joint modeling of geophysical data such as electromagnetic and gravity [6, 7]. In fact, simultaneous optimization of multiobjective functions is also favored to increase uniqueness of model parameters in joint modeling of geophysical data that are generally sensitive to different physical phenomena. Multiobjective functions can be transferred into single-objective by combination of objective functions by using weighted-sum approach. However, it is very difficult to find reasonable and optimum weigting coefficients [5]. In engineering problems, subjective and unpredictable weightings used to objective functions are the primary cause of a misleading solution, because different sensitivities and unpredictable noise of different data sets lead to uncertainty in weighting. Pareto optimality approach is a good way to obtain set of possible solutions including an optimum solution in objective function space overcoming the use of weighting and combining.

The purpose of this chapter is to review the literature on Pareto-based MOPSO. This chapter first gives a brief overview of the methods and approaches used in PSO and Pareto-based MOPSO and to look at how mathematical formulations and general algorithms of these optimization techniques work. In order to show the superiority of Pareto-based MOPSO over weighting-sum approaches, the chapter proceeds as joint modeled of two synthetic seismological data using Pareto-based MOPSO and analyses the obtained results. The results demonstrate that Pareto-based MOPSO is a useful approach to joint modeling of seismological data as explain in detail in our previous paper [8], of which TÜBİTAK is the publisher. Findings validate the usefulness of MOPSO as a technique to optimize objective functions simultaneously without weighting requirements. Finally, conclusion section gives a brief summary of MOPSO and critique of findings in modeling.

2. Particle swarm optimization

PSO method, inspired by social behavior of the bird or fish flock to reach to target in a shortest route was originally introduced by [9]. It was noticed that members of flocks suddenly change their movements as scattering and regrouping, when trajectory of swarms was observed. A striking feature of that was an effort of members to reduce their distance from both flock and surrounding members. It was found that knowledge within a flock was continuously shared by all members. PSO method was developed by defining each member in a flock as a particle. According to PSO, particles bearing an information of decision variable or model parameters take a position in an objective function space. Each particle is in communication and learning with other particles as schematically illustrated in **Figure 1a**. If a minimization problem is considered, each particle changes its position with a velocity vector and converges a global minimum as shown in **Figure 1b**.

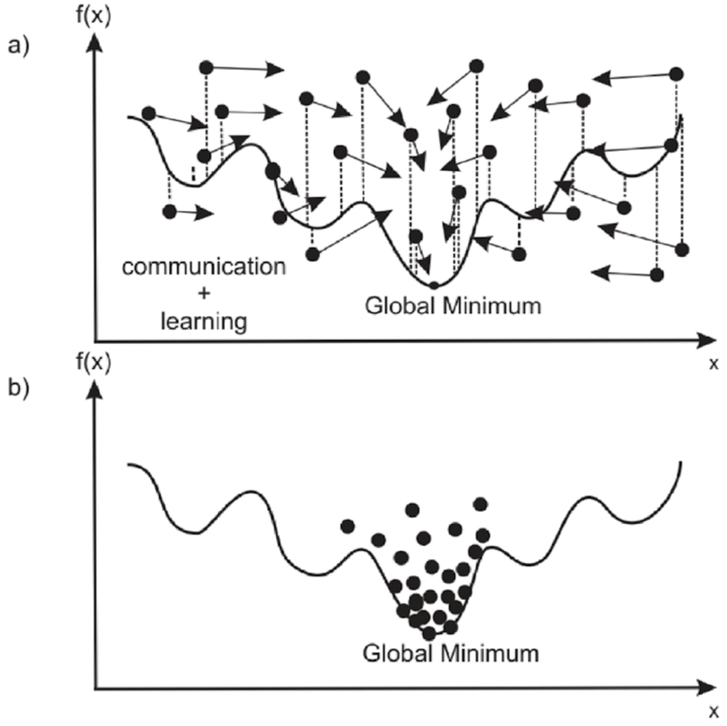
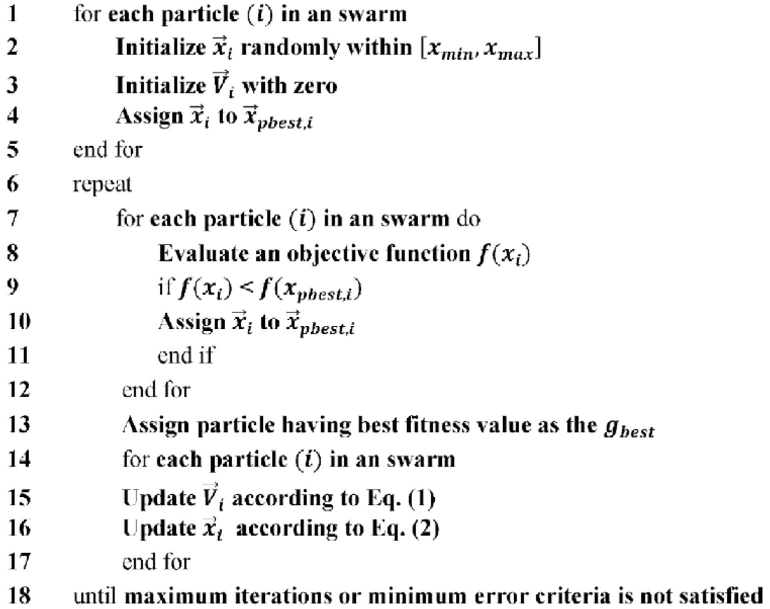


Figure 1.
For a minimization problem, schematic illustration of a swarm trajectory. Randomly distributed particles in an objective function space (a); trajectory of a swarm towards a global best solution (b), where x and f(x) denote a decision variable and an objective function, respectively.

The basic algorithm of PSO is outlined in **Figure 2**. According to this scheme, particles which velocities are initially assigned as zero are initiated by randomly selection between the minimum and maximum value of decision variables. After



```
for each particle ($i$) in an swarm
    Initialize $\vec{x}_i$ randomly within $[x_{min}, x_{max}]$
    Initialize $\vec{V}_i$ with zero
    Assign $\vec{x}_i$ to $\vec{x}_{pbest,i}$
end for
repeat
    for each particle ($i$) in an swarm do
        Evaluate an objective function $f(x_i)$
        if $f(x_i) < f(x_{pbest,i})$
        Assign $\vec{x}_i$ to $\vec{x}_{pbest,i}$
        end if
    end for
    Assign particle having best fitness value as the $g_{best}$
    for each particle ($i$) in an swarm
    Update $\vec{V}_i$ according to Eq. (1)
    Update $\vec{x}_i$ according to Eq. (2)
    end for
until maximum iterations or minimum error criteria is not satisfied
```

Figure 2.
General PSO algorithm.

each particle is evaluated with an objective function, particle with best fitness value is assigned as a global best solution. Particles move to the next position with a velocity vector

$$\vec{V}_i^{k+1} = \omega\vec{V}_i^{k} + c_1\gamma_1 \otimes \left(\vec{x}_{pbest,i} - \vec{x}_i^{k}\right) + c_2\gamma_2 \otimes \left(\vec{x}_{gbest} - \vec{x}_i^{k}\right), \tag{1}$$

where,

$$\vec{x}_i^{k+1} = \vec{x}_i^{k} + \vec{V}_i^{k+1}, \tag{2}$$

subscript i is the number of particles and k is the number of iterations. Position and velocity vector of a particle i at iteration k are shown as $\vec{x}_i^{k}$ and $\vec{V}_i^{k}$, respectively. ω is the inertia weight term forced on velocity vector. c_1 and c_2 are local and global learning constants, γ_1 and γ_2 are uniformly random numbers in the range $[0,1]$. $\vec{x}_{pbest,i}$ is the best position of particle i in the past. $\vec{x}_{gbest}$ called as leader of a swarm is the position of the best particle indicating the best fitness value. If one particle position updated by a new velocity vector has a fitness value better than its previous best, new position is assigned as $\vec{x}_{pbest}$. If one of the particles has a best fitness value than the others, it is assigned as $\vec{x}_{gbest}$. These processes are repeated in a balance between exploration and exploitation until maximum iterations or minimum error criteria is not satisfied.

2.1 Selection of the PSO parameters

Velocity vector in Eq. (1) is controlled in the following factors: velocity limitation, learning coefficients and inertia weight. These factors are significantly contribute to prevent explosion in a swarm and ensure convergence.

Velocity limitation: In the PSO method, each particle changes its velocity vector by stochastic process. However, this leads a tendency exploding of velocity vector and exceeding limit of one particle constrained in a search space, especially if the particle is far from the personal and global best position. Therefore velocity limit approach, introduced by [10], has been exploited to avoid such problems. By this limitation of movement, it is ensured that particle does not exceed a search space. Fan and Shi [11] suggested that maximum and minimum velocity limits as follows: $V_{max\,,min} = \pm U$, where $U = (x_{max} - x_{min})/N$. x_{max} and x_{min} are the upper and lower limit of a particle. N is an interval number defined by user considering the x_{max} and x_{min}.

Learning coefficients: Local and global learning coefficients illustrated in Eq. (1) control acceleration of a particle. While local learning coefficient (c_1) enables particle to approach its individual best position, global learning coefficient (c_2) enables to be pulled towards the global best solution. These parameters sometimes called as acceleration coefficients are important to control the convergence. Higher learning coefficients can provide a rapid convergence, but also probably prevents exploration in an objective function space. In [12], author suggested that dynamic learning coefficients for optimization with a large number of unknown parameters. In addition, Ozcan and Mohan [13] concluded that if $c_1 + c_2 > 4$, trajectory of a swarm goes to infinity.

Inertia weight or constriction factor: Inertia weight term (ω) introduced by Shi and Eberhart [14] maintains a balance between exploitation and exploration by acting upon the current velocity vector. They proposed that linearly decreasing inertia weight from 0.9 to 0.4 in each iteration to increase performance of the algorithm. If $\omega > 1$ is selected, particles tend to move away through boundary of a search space as velocity vectors increase continuously [5]. In [15], the author observed that particles indicate a stable and reasonable searching in case of $c_1 + c_2 \leq 4$. Nevertheless, it has even suggested a velocity limit by [16] to prevent drastically growing of velocity vector. In a major advance, Clerc [17] proposed a constriction factor (χ) to the PSO velocity vector equation. PSO algorithm with a constriction factor allowed us to rapid and reliable convergence without velocity limit requirements in case that $\phi = \phi_1 + \phi_2 > 4$, where ϕ_1 and ϕ_2 are the learning coefficients substituted with c_1 and c_2. The velocity vector is changed in the following way:

$$\vec{V}_i^{k+1} = \chi \left[\vec{V}_i^{k} + \phi_1 \gamma_1 \otimes \left(\vec{x}_{pbest,i} - \vec{x}_i^{k} \right) + \phi_2 \gamma_2 \left(\vec{x}_{gbest} - \vec{x}_i^{k} \right) \right]. \qquad (3)$$

Eq. (3) reveals that χ influences not only current velocity, but also new velocity vector. Theoretical studies in [18] show that if ϕ_1 and ϕ_2 used as 2.05, therefore ϕ = 4.1, and χ = 0.7298, rapidly convergence is observed without using velocity limits. On the other hand, in [2] the authors analyzed that if ϕ_1 and ϕ_2 are used as randomly, swarm trajectory behaves as a combination of divergence and convergence.

2.2 Swarm topologies

Network of particles in a swarm is provided by kind of neighborhood topologies that regulate sharing of information between particles. Small-scale topologies are selected to use in solving complex problems, whereas large-scale topologies are selected for simpler problem [19]. Empty, local best, fully connected, star network and tree network are the list of the neighborhood network topologies which are generally used in PSO [20].

Empty graph: Each particle is connected to itself and compared with personal best position. In this topology, it is considered that c_2 as equal to zero.

Local best: Local best $\left(\vec{x}_{lbest}\right)$ indicates that the best particle position between one particle and its nearest neighbors. In this topology, particles are affected by both their personal best and local best which is also defined as equivalent to a leader. Each particle is connected to k nearest neighbors and if k = 2, network structure of particles is called ring topology as shown in **Figure 3a**.

Fully connected graph: All particles in a swarm are connected to one another as illustrated in **Figure 3b**. In fully connected topology also called as star topology [21], each particle is affected by both its personal best and a global best defined as equivalent to leader. Kennedy [22] analyzed that this structure tends to ensure that a rapidly, but also a premature convergence.

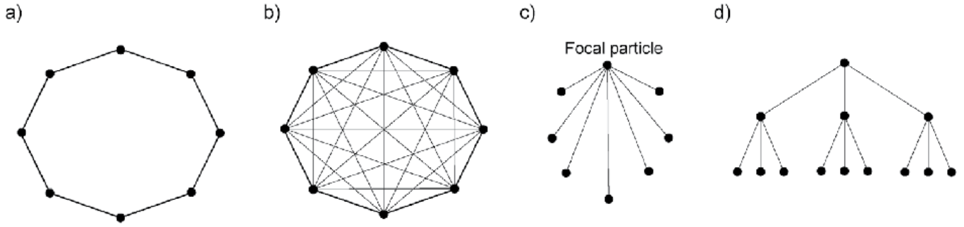


Figure 3.
Neighborhood topologies generally used in PSO: Ring (a), fully connected (b), star network (c) and tree network topology (d).

Star network: In this topology, one particle called as focal particle [23] is in connection with others as shown in **Figure 3c**. Focal particle defined as equivalent to a leader adjusts trajectory of a swarm by comparing positions of others.

Tree network: In this network structure, all particles are taken shape of a tree and, each of nodes of tree consists of exactly one particle as illustrated in **Figure 3d** [24]. One particle at the node is connected to both child nodes bottom in a tree and a parent node directly above in a tree. If one of the child node particle has a solution better than its parent solution, position of both particles are replaced. In this topology, parent particle is defined as equivalent to a leader.

3. Pareto multiobjective optimization

In multiobjective optimization problems (MOOP) indicated as an optimization of more than one objective function, "trade-off" solutions that are conflicted to each other are obtained rather than single solution. MOOP is generally defined to obtain decision variables

$$x = (x_1, x_2, x_3, \dots \dots \dots x_p) \in S \tag{4}$$

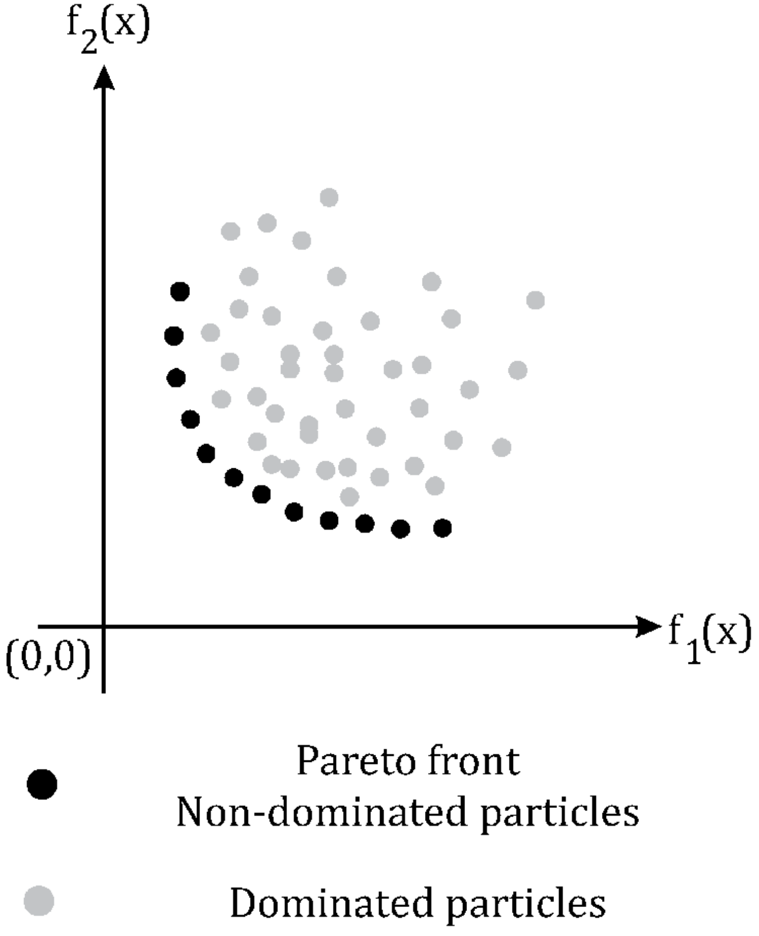


Figure 4.
Conceptual representation of the non-dominated and dominated particles in objective function space. Here, $f_1(x)$ and $f_2(x)$ are objective functions. [8].

in p dimensional of model space S, while simultaneously optimizing of a vector

$$f(x) = \left[f_1(x), f_2(x), f_3(x) \dots \dots f_N(x) \right] \tag{5}$$

including all objective functions [25]. In MOOP, all objective functions in $f(x)$ vector may not be optimized simultaneously. Therefore Pareto optimality sometimes called as Pareto dominance approach is used to find set of possible solutions. According to Pareto approach, if x is better solution than y in minimization problem, $f(x)$ dominates $f(y)$, if and only if $f_k(x) \leq f_k(y), k = 1, \dots ., N$. If there is no other $\overline{x} \in S$ satisfies a condition such that $f(\overline{x}) < f(x)$ without deteriorating any other objective function, non-dominated solutions called Pareto optimal set $P^* \in S$ or Pareto front $PF^* = \{ f(x) \in \mathbb{R} \mid x \in P^* \}$ exists in an objective function space as schematically illustrated in **Figure 4**.

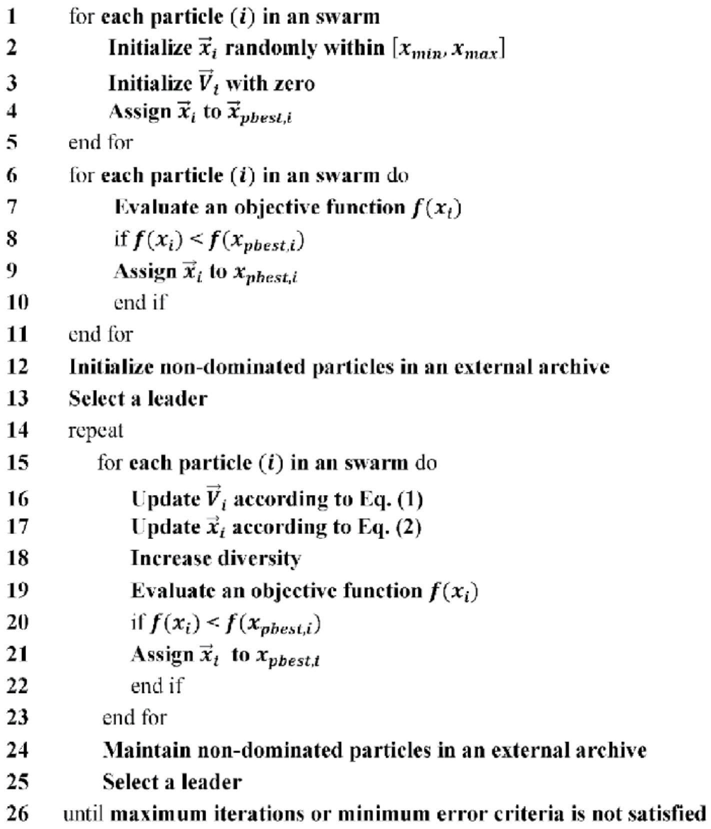
All particles of Pareto front generally spreads in two ways which are concave and L-shaped curves. Concave shaped spreading indicates that one objective function does not be ameliorated without deteriorating any other objective function(s). On the other hand, L-shaped spreading indicates that complete optimal solution which means that all objective functions can be optimized simultaneously [26]. Other remarkable distribution of Pareto front is its deviation from symmetry referenced by utopia point [0,0]. Deviation is an indication that one objective function has many local minimums relatively the others (minimum one or more) and/or modeling has not properly accomplished with defined parameter search space and used methods [27].

4. Pareto-based multiobjective particle swarm optimization

Several considerations that are increasing the diversity, maintaining the non-dominated particles and selecting a leader should be taken into account when using MOPSO. [20] A general MOPSO algorithm modified from PSO algorithm by these considerations outlined below is shown in **Figure 5**.

Increasing of diversity: In MOPSO, it is expected to increase of diversity of all particles, while dominated particles are required to converge towards to Pareto-front. Mutation, niching and clustering techniques are generally used to increase diversity. Mutation operator is originally a stage of the genetic algorithm and it prevents to trap of particles a local minima by altering decision variables within the bounds of possibility [28, 29]. suggested that polynomial and dynamic mutation rate which should be changed from higher to lower degree to keep increasing diversity in an entire search space. This feature allows particles to be explored, even in far field on this space [30]. Niching method, which was also developed for evolutionary algorithms [31], prevents a premature convergence that is a leading cause of a movement of particles towards a solution [32]. In NichePSO presented by [33], alteration of each particle is kept track after each iteration. If some particles do not change after several iterations, niche formed as a sub-swarm including both these particles and their surroundings is generated with a radius called σ_{share} as illustrated in **Figure 6**. By comparing of two particles in niches, a control mechanism is provided to be adding particles to a sub-swarm or to be merging of sub-swarms [34]. In clustering technique introduced by [35], main swarm is divided into k sub-swarms and determined leader in each sub-swarm is moved to centroid of its sub-swarm. Diversity is provided by a migration of leaders to different sub-swarms [36].

Maintaining non-dominated particles: Maintaining non-dominated particles is key component of MOPSO. External archive technique allowing a storage of non-dominated particles is generally used to spread and maintain Pareto-front. Global



```
for each particle ($i$) in an swarm
    Initialize $\vec{x}_i$ randomly within $[x_{min}, x_{max}]$
    Initialize $\vec{V}_i$ with zero
    Assign $\vec{x}_i$ to $\vec{x}_{pbest,i}$
end for
for each particle ($i$) in an swarm do
    Evaluate an objective function $f(x_i)$
    if $f(x_i) < f(x_{pbest,i})$
    Assign $\vec{x}_i$ to $x_{pbest,i}$
    end if
end for
Initialize non-dominated particles in an external archive
Select a leader
repeat
  for each particle ($i$) in an swarm do
      Update $\vec{V}_i$ according to Eq. (1)
      Update $\vec{x}_i$ according to Eq. (2)
      Increase diversity
      Evaluate an objective function $f(x_i)$
      if $f(x_i) < f(x_{pbest,i})$
      Assign $\vec{x}_i$ to $x_{pbest,i}$
      end if
    end for
    Maintain non-dominated particles in an external archive
    Select a leader
until maximum iterations or minimum error criteria is not satisfied
```

Figure 5.
General MOPSO algorithm.

best, personal best or local best can be stored to an archive when extending PSO to MOPSO. However, as dominated particles tending to converge and join to Pareto-front, a rapid increase number of particles in an archive is a disadvantage [20]. A restriction is applied to prevent rapidly growing number of non-dominated particles. In [37], they proposed ϵ-dominance value defined as set of boxes in objective function space as shown in **Figure 7** controls adding of dominated particles to Pareto-front by sizing of an archive. In [38], their attempts to update of external archive is composed niching method used to add or remove particles from archive when it is full. In [30], geographically-based global repository approach is used to update of an archive based on removing of non-dominated solutions from cells which have fewer particles. Li [39] proposed non-dominated sorting method to determine Pareto-front in each iteration. In his study, rather than comparing of a new position with a personal best, temporary swarms have been generated by combining both of them. In fact, the fundamental condition maintaining non-dominated particles is that a new archive should be dominant over previous archives. ϵ-dominance approach is one of the most feasible method to provide this condition [20].

Leader selection: Unlike one leader that guides a swarm in the PSO, non-dominated particles of Pareto front set indicate presence of multiple leaders in MOPSO. Neighbor density and kernel density estimator methods based on density measurement techniques are suggested to determine particles that are most likely to be selected as a leader. In [40], they studied and further supported the concept of crowding factor according to nearest neighbors of a given particle. The value of

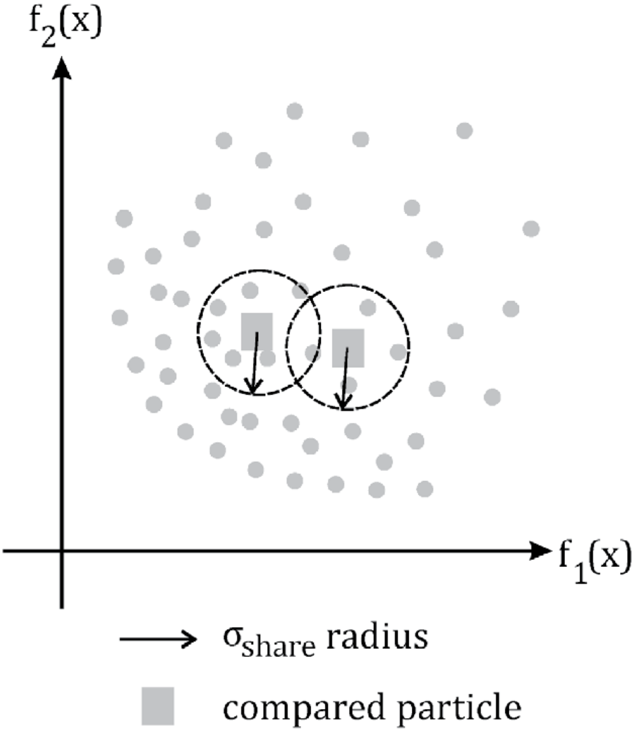


Figure 6.
Niches having radius indicated by σ_{share} are generated to compare of particles that have not change after several iterations.

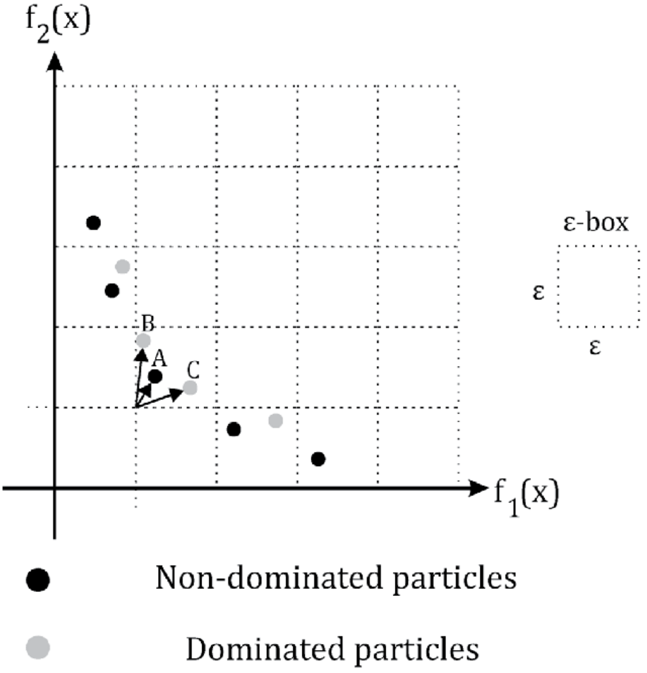


Figure 7.
ϵ-dominance concept for minimizing two objective functions. Particle A, B and C are incommensurable particles, however, particle a is considered to be dominant to B and C due to more closer the lower left hand corner.

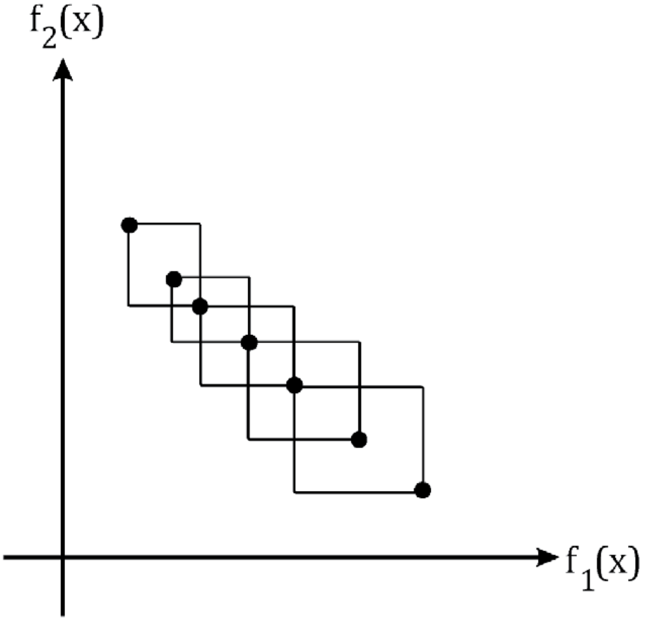


Figure 8.
Neighbor density estimator defined by perimeter of rectangular formed between one particle and its neighbors.

estimators are determined by perimeter of a rectangular formed between one particle and neighbors as illustrated in **Figure 8**. Particles which have larger value of estimator are preferred to select as a leader.

The kernel density estimator is based on the niching method. A niche which has radius as σ_{share} is generated for each of non-dominated particles. It is preferable selected a leader from the least crowded niche [41]. As for leader selection from preferred area, different ways are used. For instance, neighbor density estimator has been used by Raquel and Naval [42] to select a leader randomly from top a list ranking in descending order according to estimator values. If an external archive is full, particle has been removed by randomly selecting from bottom of the list. To select a leader, Coello Coello et al. [30] suggested to fitness values based on the number of particles in each hypercube formed by division of an objective function space. A leader is randomly selected from a hypercube determined by roulette-wheel selection scheme which its lengths of divided segments proportional to fitness values.

5. Selecting the methods and parameters for Pareto-based multiobjective particle swarm optimization

We joint modeled two synthetic seismological data obtained by response of five-layered models by using MOPSO. In one synthetic model, shear wave velocities increase smoothly with depth (SM-1), while the other has a noticeable velocity contrast at third layer interface (SM-2). In the modeling stage, we simultaneously optimized two objective functions related with Rayleigh wave dispersion (RWD) and horizontal to vertical spectral ratio (HVSR) methods, which have different sensivities to physical phenomena. The estimated parameters were shear-wave velocities and depth obtained by cumulative sum of layer thicknesses in each layer. Parameter search space and more technical details were be given in [8].

Minimization was carried out between observed data obtained by response of synthetic models (y_i^{obs}) and calculated data (y_i^{cal}) obtained by response of models which their parameters changing with particle movements. Fitness value (α) minimized at each of the objective function was calculated using,

$$\alpha = \left[\frac{1}{n}\sum_{i=1}^{n}\left(y_i^{obs} - y_i^{cal}\right)^2\right]^{1/2}, \tag{6}$$

where $\boldsymbol{n}$ is the number of observations. Number of iterations and particles was selected as 200 and 100, respectively. Number of particles was selected as 100 referenced by 10-fold the number of model parameters (thicknesses and velocities for 5 layers). We used PSO parameters defined in [43] with velocity equation in Eq. (3) which provides an optimization without a velocity limitation. A mutation operator was used to increase diversity and mutation rate was selected as two percent. ϵ-dominance approach was selected to maintain of non-dominated particles and the value of ϵ was used as 0.01 for each of the objective functions. Number of hypercubes was selected as 30 for 100 particles recommended by [30].

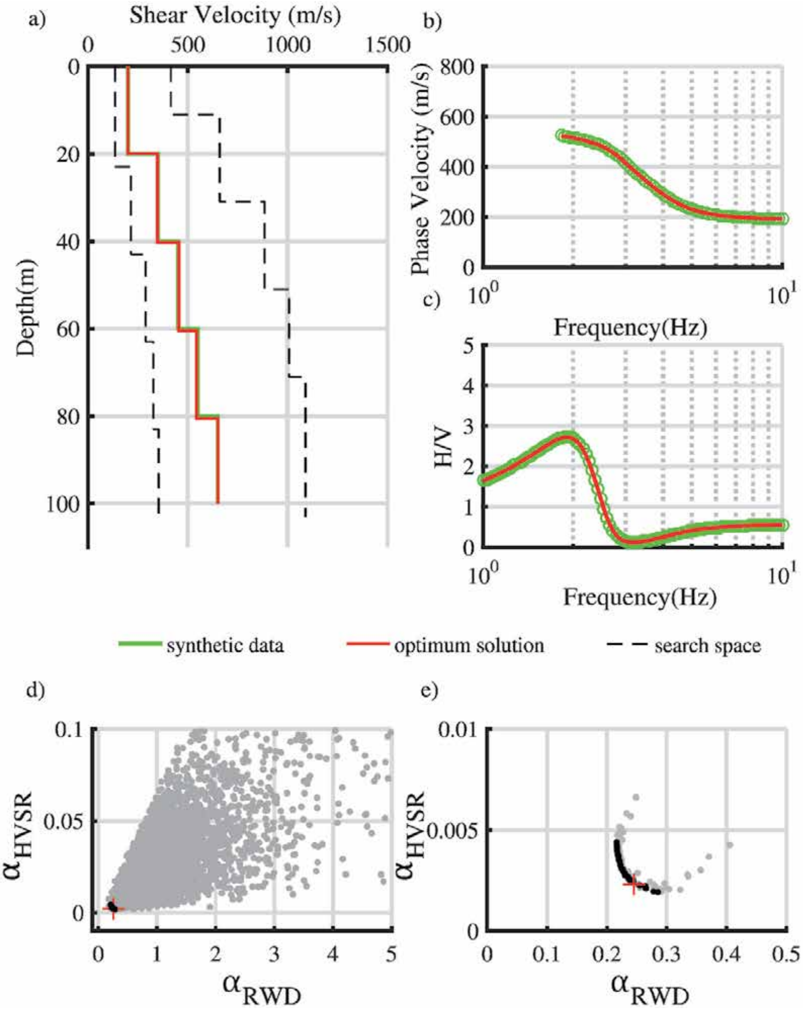


Figure 9.
Results for SM-1. Synthetic model and a model obtained from the Pareto optimum particle and the parameter search space (a); the fit between the observed and calculated RWD (b), and HVSR (c); the Pareto optimum solution marked as + and Pareto front (dark dots) with the dominated particles (light dots) for all iterations (d); and the Pareto optimum solution (+) with Pareto front and dominated particles at the last iteration (e) [8].

Hypercubes having more particles were preferred for leader selection. After a fitness hypercube was determined by using Roulette-wheel approach, a leader was selected randomly from this hypercube.

6. Results and discussions

The results of synthetic models (SM-1 and SM-2) are shown in **Figure 9** and **Figure 10**, respectively. As can be seen in **Figure 9a** and **Figure 10a**, MOPSO was successful in proving to obtain the real models with defined methods and parameters. These tests showing matched perfectly between synthetic data and model

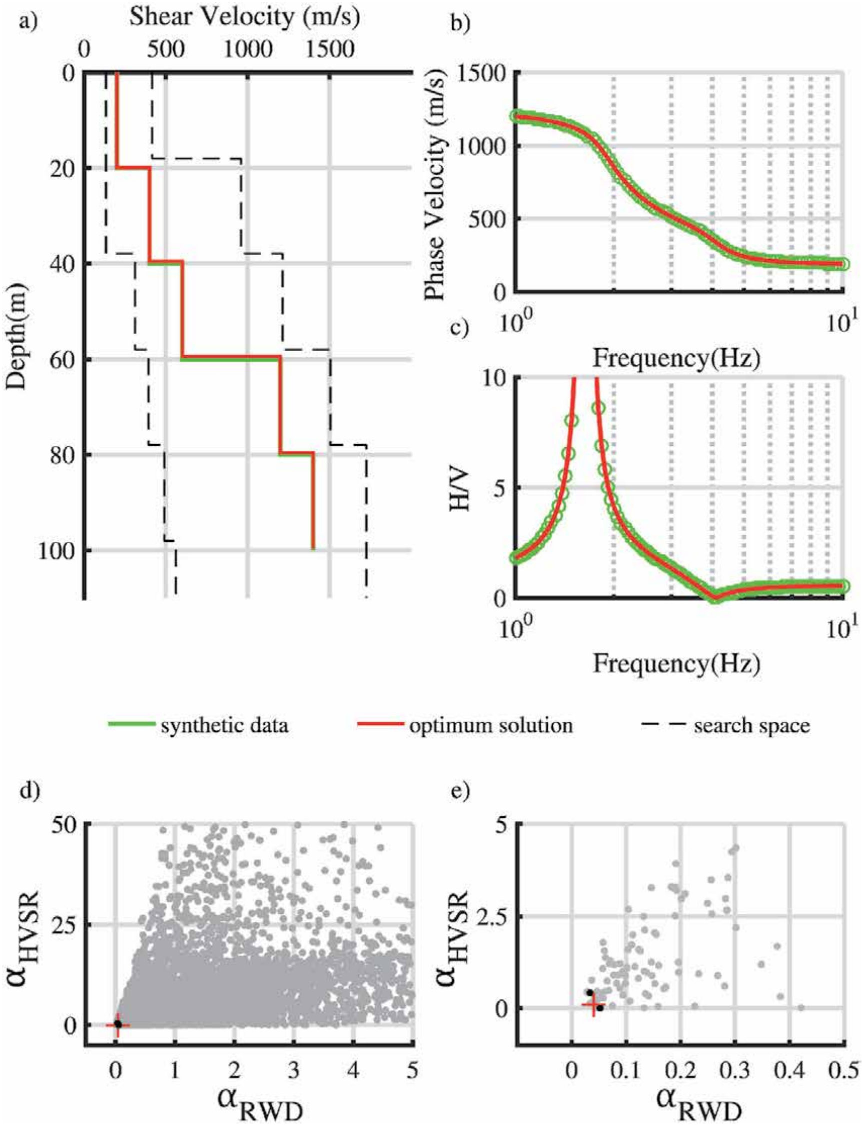


Figure 10.
Results for SM-2. Synthetic model and a model obtained from the Pareto optimum particle and the parameter search space (a); the fit between the observed and calculated RWD (b), and HVSR (c); the Pareto optimum solution marked as + and Pareto front (dark dots) with the dominated particles (light dots) for all iterations (d); and the Pareto optimum solution (+) with Pareto front and dominated particles at the last iteration (e) [8].

responses as seen in **Figure 9b**, **Figure 9c**, **Figure 10b** and **Figure 10c**, further strengthened our confidence in MOPSO modeling such seismological data. The most conspicuous observation to emerge from the result of both models was the distribution of dominated particles and Pareto front. Dominated particles in **Figure 9d** show a clear and balanced distribution for SM-1, however, as in **Figure 10d** dominated particles tend to towards the α_{RWD} axis for SM-2. This results showed that if only HVSR optimization was done, many local minima and non-unique solutions could be obtained in modeling of data obtained by response of such as SM-2 model. In contrast to earlier findings in [27], deviation from symmetry and divergence from an objective function axis were not only related to a improperly accomplished modeling, but also to the characteristic of HVSR optimization in case of model such as SM-2 that its optimization indicated non-uniqueness solutions. In addition, as shown in **Figure 9b**, spreading of Pareto front in SM-1 modeling showed a concave shaped curve that means an objective function could not further minimized without maximized to other objective function. However, for SM-2 model, Pareto front showed a similar distribution to L-shape, indicating that both objective functions were minimized independently and simultaneously.

7. Conclusion

This chapter considered is an overview of methods and parameters generally used in PSO and Pareto-based MOPSO, in the first step. The parameters and methods used in the literature are reliable but do not have an obvious superiority to each other. In spite of that MOPSO has been widely applied in many real world engineering problems, a few attempts have been made in order to modeling geophysical data. Until our previous study, this methodology have not been applied modeling seismological data. A set of solutions demonstrated in this chapter support the idea that MOPSO provides a powerful methodology for joint modeling of data having different sensivity. The present findings have important implications in order to solve weighting problem encountered in joint modeling approach. A clear benefit of MOPSO in the prevention of weighting-sum approaches could be clearly identified in this analysis. A further important implication is that divergence of particles from an objective function axis is not only related to properly defined parametrization and accomplished modeling, but also to non-uniqueness solutions. Our investigations into this point are still ongoing and seem likely to confirm our hypothesis. The evidence from this study implies that MOPSO is considered to be very attractive for joint modeling geophysical data in the future. However, further work needs to be performed to confirm whether MOPSO is beneficial to joint modeling of different types of geophysical data.

Acknowledgements

I am grateful to Prof. Dr. Abdullah Karaman and Assoc. Prof. Dr. Ekrem Zor for their help and valuable suggestions and discussions. Support was given by Istanbul Technical University and The Council of Higher Education (YÖK), who funded the work in its initial stages.

Author details

Ersin Büyük
Department of Geophysics, İstanbul Technical University, İstanbul, Turkey

*Address all correspondence to: ebuyuk@itu.edu.tr

IntechOpen

© 2021 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

References

[1] Rao SS. Engineering Optimization: Theory and Practice. Fourth Edition. Hoboken, NJ, USA: John Wiley & Sons, Inc; 2009. 813 p. DOI: 10.1002/9780470549124

[2] Van Den Bergh F, Engelbrecht AP. A study of particle swarm optimization particle trajectories. Information Sciences. 2006; 176: 937–971. DOI: 10.1016/j.ins.2005.02.003

[3] Büyük E, Zor E, Karaman A. Rayleigh wave dispersion curve inversion by using particle swarm optimization and genetic algorithm. In: 19th European Geophysical Union EGU General Assembly; 2017; Vienna, Austria. p. 6911

[4] Engelbrecht AP. *Computational Intelligence: An Introduction. Second Edition*. John Wiley and Sons, Inc; 2007. 628 p. DOI: 10.1002/9780470512517.

[5] Blum C, Li X. Swarm Intelligence in Optimization. In: Blum C, Merkle D, editors. Swarm Intelligence Introduction and Applications. Berlin Heidelberg: Springer-Verlag; 2008. p. 43–85. DOI: 10.1007/978-3-540-74089-6_2

[6] Pace F, Godio A, Santilano A, et al. Joint optimization of geophysical data using multi-objective swarm intelligence. Geophysical Journal International. 2019; 218: 1502–1521. DOI: 10.1093/gji/ggz243

[7] Danek T, Leśniak A, Miernik K, et al. Pareto Joint Inversion of 2D magnetometric and gravity data-synthetic study. E3S Web of Conference. 2019; 133. DOI: 10.1051/e3sconf/201913301009

[8] Büyük E, Zor E, Karaman A. Joint modeling of rayleigh wave dispersion and H/V spectral ratio using pareto-based multiobjective particle swarm optimization. Turkish Journal of Earth Science. 2020; 29: 684–695. DOI: 10.3906/yer-2001-15

[9] Kennedy J, Eberhart R. Particle swarm optimization. Neural Networks. 1995; 4: 1942–1948. DOI: 10.1109/ICNN.1995.488968

[10] Eberhart RC, Shi Y. Particle swarm optimization: Developments, applications and resources. In: Proceedings of the IEEE Conference on Evolutionary Computation, ICEC; 2001; 1: 81–86. DOI: 10.1109/cec.2001.934374

[11] Fan H, Shi Y. Study on Vmax of particle swarm optimization. In: Proceedings of the Workshop on Particle Swarm Optimization. Indianapolis: Purdue School of Engineering and Technology, IUPUI. 2001

[12] Ratnaweera A, Halgamuge SK, Watson HC. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on Evolutionary Computation. 2004; 8: 240–255. DOI: 10.1109/TEVC.2004.826071

[13] Ozcan E, Mohan CK. Particle swarm optimization: Surfing the waves. In: *Proceedings of the 1999 Congress on Evolutionary Computation, CEC; 1999*; pp. 1939–1944. DOI: 10.1109/CEC.1999.785510

[14] Shi Y, Eberhart R. Modified particle swarm optimizer. In: *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*; 1998; pp. 69–73. DOI: 10.1109/icec.1998.699146

[15] Kennedy J. The behavior of particles. In: Porto VW, Saravanan N, Waagen D, editors. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

Berlin: Springer Berlin Heidelberg; 1998. p. 581–589. DOI: 10.1007/bfb0040809

[16] Venter G, Sobieszczanski-Sobieski J. Particle Swarm Optimization. AIAA Journal. 2003; 41: 1583–1589. DOI: 10.2514/2.2111

[17] Clerc M. The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC; 1999; pp. 1951–1957. DOI: 10.1109/ CEC.1999.785513

[18] Eberhart RC, Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. In: *Proceedings of the 2000 Congress on Evolutionary Computation, CEC; 2000*; pp. 84–88. DOI: 10.1109/ CEC.2000.870279

[19] Mendes R, Kennedy J, Neves J. The fully informed particle swarm: Simpler, maybe better. IEEE Transactions on Evolutionary Computation. 2004; 8: 204–210. DOI: 10.1109/ TEVC.2004.826074

[20] Coello Coello CA, Reyes-Sierra M. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. International Journal of Computational Intelligence Researchs. 2006; 2: 287–308. DOI: 10.5019/j. ijcir.2006.68

[21] Eberhart RC, Shi Y. Computational Intelligence. In: Eberhart RC, Shi Y, editors. Morgan Kaufmann; 2007; p. 17–38. DOI: 10.1016/B978-155860759-0/50002-0

[22] Kennedy J. Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999. IEEE Computer Society; 1999; pp. 1931–1938. DOI: 10.1109/CEC.1999.785509

[23] Engelbrecht AP. Computational Intelligence: An Introduction. John Wiley & Sons; 2005. p. 597. DOI: 10.1002/9780470512517

[24] Janson S, Middendorf M. A hierarchical particle swarm optimizer. In: 2003 Congress on Evolutionary Computation, CEC 2003 - Proceedings; IEEE Computer Society; 2003. pp. 770–776. DOI: 10.1109/CEC.2003.1299745

[25] Deb K. Multi-Objective Optimization Using Evolutionary Algorithms. New York USA: John Wiley & Sons; 2001. p 518.

[26] Kozlovskaya E, Vecsey L, Plomerova J, et al. Joint inversion of multiple data types with the use of multiobjective optimization: problem formulation and application to the seismic anisotropy investigations. *Geophysical Journal International.* 2007; 171: 761–779. DOI: 10.1111/ j.1365-246X.2007.03540.x

[27] Dal Moro G. Insights on surface wave dispersion and HVSR: Joint analysis via Pareto optimality. *Journal of Applied Geophysics.* 2010; 72: 129–140. DOI: 10.1016/j.jappgeo.2010.08.004

[28] Angeline PJ. Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In: Porto VW, Saravanan N, Waagen D, et al., editors. Evolutionary Programming VII. Berlin: Springer, Berlin, Heidelberg; pp. 601–610. DOI: 10.1007/bfb0040811

[29] Van Den Bergh F. An Analysis of Particle Swarm Optimizers. PhD Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2006; p. 284.

[30] Coello Coello C, Pulido GT, Lechuga MS. Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computationrans. 2004; 8:

256–279. DOI: 10.1109/TEVC.2004.826067

[31] Goldberg DE, Holland JH. Genetic Algorithms and Machine Learning. *Machine Learning;* 1988; 3: 95–99. DOI: 10.1023/A:1022602019183

[32] Van den Bergh F, Engelbrecht AP. A new locally convergent particle swarm optimiser. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics. 2002, pp. 94–99. DOI: 10.1109/icsmc.2002.1176018

[33] Brits R, Engelbrecht AP, Bergh F Van Den. A Niching Particle Swarm Optimizer. In: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL 2002). Orchid Country Club, Singapore, 2002, pp. 692–696.

[34] Barrera J, Coello CAC. A review of particle swarm optimization methods used for multimodal optimization. In: Chee Peng Lim, Lakhmi C. Jain, Dehuri S, editors. *Inniovations in Swarm Intelligence*. 2009, pp. 9–37. DOI: 10.1007/978-3-642-04225-6_2

[35] Kennedy J. Stereotyping: Improving particle swarm performance with cluster analysis. In: *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*. 2000, pp. 1507–1512.

[36] Pulido GT, Coello Coello CA. Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer. In: Deb K, editor. Genetic and Evolutionary Computation. Lecture Notes in Computer Science: 2004; 3102: 225–237. DOI: 10.1007/978-3-540-24854-5_20

[37] Laumanns M, Thiele L, Deb K, et al. Combining convergence and diversity in evolutionary multiobjective optimization. Evolutionary Computation. 2002; 10: 263–282. DOI: 10.1162/106365602760234108

[38] Salazar-Lechuga M, Rowe JE. Particle swarm optimization and fitness sharing to solve multi-objective optimization problems. In: Congress on Evolutionary Computation (CEC'2005). 2005; Edinburgh, Scotland, UK: IEEE Press; 2: 1204–1211. DOI: 10.1109/cec.2005.1554827

[39] Li X. A non-dominated sorting particle swarm optimizer for multiobjective optimization. In: Cantú-Paz E. et al., editors. Genetic and Evolutionary Computation. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg: 2003; pp. 37–48. DOI: 10.1007/3–540-45105-6_4

[40] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. In: IEEE Transactions on Evolutionary Computation. 2002; 6:182–197. DOI: 10.1109/4235.996017.

[41] Deb K, Goldberg DE. An investigation of niche and species formation in genetic function optimization. In: Proceedings of the third international conference on Genetic algorithms. 1989;pp. 42–50.

[42] Raquel CR, Naval PC. An effective use of crowding distance in multiobjective particle swarm optimization. In: GECCO 2005 - Genetic and Evolutionary Computation Conference. New York, New York, USA: ACM Press, 2005; pp. 257–264. DOI: 10.1145/1068009.1068047

[43] Kennedy J. Swarm Intelligence. In: Zomaya A.Y. editor. Handbook of Nature-Inspired and Innovative Computing. Springer, Boston: 2006; pp 187–219. DOI: 10.1007/0–387-27705-6_6

Edited by Nodari Vakhania
and Mehmet Emin Aydin

Optimisation is one of the unavoidable key subjects in engineering and other real-world problems, which attracts researchers' and practitioners' attention for decades. On the other hand, computational algorithms nowadays play a definitive role in most real-life applications, from mobile phones to supercomputers, Internet servers, manufacturing, etc. An intelligent method for the enumeration of feasible solutions may lead to efficient computational algorithms. Swarm intelligence emerges as a rather new and novel of field computational intelligence that turned into a hot spot in optimization studies last two decades.

This book brings together a number of research articles within the intersection of these two prominent subjects, which introduces techniques and approaches in detail and demonstrates how optimisation problems can be solved with heuristic and swarm intelligence approaches. It contains a few contributions on Particle Swarm Optimisation (PSO) area, which is one of renown swarm optimisation approaches that will shed light to issues around optimisation with swarm intelligence to guide junior researchers with implementation details provided.

Published in London, UK

© 2022 IntechOpen
© carloscastilla / iStock

IntechOpen

ISBN 978-1-83968-667-2
9 781839 686672