

25(3) 2024

ISSN 2300-7036

COMPUTER SCIENCE

AGH UNIVERSITY PRESS

KRAKOW 2024

Editor-in-chief: *Jacek Kitowski*, AGH University of Krakow

Co-editors

Andrzej Bielecki
AGH University of Krakow

Piotr Kulczycki
AGH University of Krakow

Marek Kisiel-Dorohinicki
AGH University of Krakow

Konrad Kułakowski
AGH University of Krakow

Piotr A. Kowalski
AGH University of Krakow

Kazimierz Wiatr
AGH University of Krakow, ACC Cyfronet AGH

Assistant editors

Aleksander Byrski
AGH University of Krakow

Radosław Łazarz
AGH University of Krakow

Editorial Board

Stanisław Ambroszkiewicz
Polish Academy of Sciences

Krzysztof Boryczko
AGH University of Krakow, Poland

Jeffrey M. Bradshaw
Institute for Human and Machine Cognition, USA

Piotr Breitkopf
Universite de Technologie de Compiegne, France

Peter Brezany
University of Vienna, Austria

Marian Bubak
AGH University of Krakow, Poland,
University of Amsterdam, Netherlands

Tadeusz Burczyński
Silesian University of Technology, Poland

Marco Carvalho
Florida Institute of Technology, United States

Krzysztof Cios
Virginia Commonwealth University, USA

Carlos Cotta
University of Malaga, Spain

Paweł Czarnul
Gdansk University of Technology, Poland

Ireneusz Czarnowski
Gdynia Maritime University, Poland

Ewa Deelman
University of Southern Carolina, USA

Leszek Demkowicz
University of Texas in Austin, USA

Grzegorz Dobrowolski
AGH University of Krakow, Poland

Marco Dorigo
Université Libre de Bruxelles, Belgium

Andrzej Duda
INPG, France

Witold Dzwiniel
AGH University of Krakow, Poland

Piotr Faliszewski
AGH University of Krakow, Poland

Vladimir Getov
University of Westminster, UK

Andrzej M. Gościnski
Deakin University, Australia

Jerzy W. Grzymała-Busse
University of Kansas, USA

Ladislav Hluchy
Slovak Academy of Sciences

Bipin Indurkha
International Institute of Information Technology, India

Janusz Kacprzyk
Systems Research Institute, Polish Academy of Sciences

Joanna Kołodziej
Cracow University of Technology, Poland

Zdzisław Kowalczyk
Gdansk University of Technology, Poland

Dieter Kranzlmüller
Ludwig-Maximilians-Universität, Germany

Piotr Łuszczek
University of Tennessee, USA

Stan Matwin
University of Ottawa, Canada

Zbigniew Michalewicz
University of Adelaide, Australia

Pablo Moscato
The University of Newcastle, Australia

Grzegorz Jacek Nalepa
AGH University of Krakow, Poland

Marek R. Ogiela
AGH University of Krakow, Poland

Maciej Paszyński
AGH University of Krakow, Poland

Witold Pedrycz
University of Alberta, Canada

Juan Carlos Burguillo Rial
University of Vigo, Spain

Muzafer H. Saračević
Department of Computer Sciences, University
of Novi Pazar, Serbia

Andrzej Skowron
University of Warsaw, Poland

Marcin Szpyrka
AGH University of Krakow, Poland

Vilem Srovnal
Technical University of Ostrava, Czech Republic

Bolesław Szymański
Academic Research Center for Social
and Cognitive Networks RPI, USA

Ryszard Tadeusiewicz
AGH University of Krakow, Poland

Marek Tudruj
Institute of Computer Science, Polish Academy of Sciences,
Poland

Gabriele von Voigt
University of Hannover, Germany

Katarzyna Węgrzyn-Wolska
ESIGETEL, France

Stefan Wesner
Communication and Information Centre
University of Ulm, Germany

Janusz Wojtusiak
George Mason University, US

Julius Zilinskas
Vilnius University, Lithuania

25(3) 2024

ISSN 2300-7036

COMPUTER SCIENCE



AGH UNIVERSITY PRESS

KRAKOW 2024

EDITORIAL INFORMATION

Editor-in-Chief

Jacek Kitowski

Co-Editors

Andrzej Bielecki

Marek Kisiel-Dorohinicki

Piotr A. Kowalski

Piotr Kulczycki

Konrad Kułakowski

Kazimierz Wiatr

Assistant Editors

Aleksander Byrski

Radosław Łazarz

COMPUTER SCIENCE is published by AGH University Press, Krakow, Poland.

The papers presented in COMPUTER SCIENCE have been accepted by the reviewers selected by the editors of the journal.

Head of Publishing of AGH University Press

Jan Sas

Technical Editor

Magdalena Grzech

Ghostwriting prevention

Łukasz Faber

Statistical Correction

Anna Barańska

Linguistic Correction

Bret Spainhour

Cover Design

Anna Sadowska

Typesetting and Desktop Publishing

Marek Karkula

© Wydawnictwa AGH, Kraków 2024

Creative Commons License Attribution 4.0 International (CC BY 4.0)

ISSN 2300-7036

DOL: <https://doi.org/10.7494/csci>

Wydawnictwa AGH (AGH University Press)

al. A. Mickiewicza 30, 30-059 Kraków, Poland

tel. +48 12 617 32 28, +48 12 636 40 38

e-mail: redakcja@wydawnictwoagh.pl

<https://www.wydawnictwa.agh.edu.pl>

CONTENTS

Krzysztof Ostrowski, Mateusz Starzec, Grażyna Starzec

The ant colony optimization algorithm applied in transport logistics. . . . 331

Saib Bouthina, Mohamed-Rida Abdessemed, Riadh Hocine

CV19T, a novel bio-socially inspired method,
belonging to a new nature-inspired metaheuristics class. 351

Chafika Djaoui, Allaoua Chaoui

Formalization and analysis of UML 2.0 interaction overview diagram
using Maude rewriting logic language. 397

Aldin Kovačević, Muzafer Saračević, Amor Hasić

Biometrics-based generation of Diffie-Hellman
key exchange parameters 421

Dheepika PS, Umadevi V

A novel hybrid deep learning approach for 3D object detection
and tracking in autonomous driving. 435

Heba F. Eid, Erik Cuevas

Enhanced Bonobo optimizer for optimizing
dynamic photovoltaic models. 469

KRZYSZTOF OSTROWSKI

MATEUSZ STARZEC

GRAŻYNA STARZEC

THE ANT COLONY OPTIMIZATION ALGORITHM APPLIED IN TRANSPORT LOGISTICS

Abstract *The Vehicle Routing Problem belongs to graph optimization and its goal is to find shortest routes visiting a given set of customers with additional constraints present. The article presents the ant colony optimization metaheuristic which solves vehicle routing problems and its real-life application in transport logistics (finding routes for delivery companies). The metaheuristic generated high-quality solutions (superior to compared methods). Our tool is flexible and enables us to solve various variants of routing problems so it is well suited to specific needs of transportation companies.*

Keywords ant colony optimization, ACO, metaheuristic, routing problems, transport logistics, delivery

Citation Computer Science 25(3) 2024: 331–350

Copyright © 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

Routing in transportation networks is a daily task for transport industry companies (i.e. transportation, delivery planning or waste collection). Constructing efficient routes can be challenging (especially for larger fleets of vehicles and with multiple additional constraints) but is necessary to minimize costs (i. e. fuel consumption and work time). Routes optimization can be regarded as a part of green logistics, which is an activity aiming to reduce the environmental impact of companies logistics. Constructing efficient routes that visit a given set of customers usually have additional constraints i. e. associated with vehicles capacity, customers opening-times or vehicles distance/work-time limit. Optimization problems arising from practical needs of transportation companies belong to the family of Vehicle Routing Problems (VRP). The problem's computational difficulty (impractical execution time of exact methods for larger instances) triggered the search for efficient metaheuristic solutions.

The authors' previous research was focused mostly on metaheuristics and graph problems [24, 27]. The authors propose the ant-colony optimization metaheuristic (ACO) as a part of a web application, which solves various problems from VRP family (dependent on the needs of transportation companies). The presented algorithm is a combination of techniques and variants of ACO and local optimizations. This is a result of comprehensive tests conducted on tens of real-life optimization tasks. We collected this data from companies that are interested in implementation of our solution in their daily work. Our work was associated with a project named "TRASA – development and validation of algorithms for routes optimization and resources allocation" [30]. The article is organized as follows: Section 2 presents VRP family, Section 3 includes mathematical formulations, Section 4 outlines some existing solutions, Section 5 describes the algorithm, Section 6 presents real-life results and conclusions are included in Section 7.

2. VRP family

The Vehicle Routing Problem is a multiple route generalization of the well-known Travelling Salesman Problem (TSP) [5], where the goal is to find a Hamiltonian cycle (a path in a graph visiting all vertices), which minimizes the total cost of visited edges. VRP is a family of combinatorial optimization problems, which has many real-life applications (i. e. those mentioned above) and dates back to 1959 [10]. In the basic version of the problem there is a graph of n customer locations and a fleet vehicle consisting of m cars. There is also one or more depots: special locations representing start and end of cars routes. The graph is weighted (costs associated with edges between locations). The goal is to visit all customers (using at most m cars) while minimizing the total cost of all routes (each route starts and ends in given depots).

VRP has many variants, which add constraints to the basic optimization problem and were defined to address practical routing problems. In Capacitated Vehicle Routing Problem (CVRP) [16] each vehicle has a limited capacity for goods that are

delivered to clients (each client has some demand) and therefore routes sizes are limited by max. capacity. In this case, number of vehicles used can be also optimized (as the first optimization criterion – the second being total routes cost). In Vehicle Routing Problem with Time Windows (VRPTW) each client has a time-window [7]. In hard TW version goods can be delivered only in this time slot: arriving too early means waiting for a TW opening while arriving too late means delivery is impossible. In VRP versions with soft time-windows (VRPSTW) delivery time bounds can be violated but at a penalty [4]. In time-window versions of the problem edge also have assigned travel times and there are also service times assigned to each client. In Distance Constrained Capacitated Vehicle Routing Problems (DCVRPs) the duration of each vehicle route (defined as the sum of travel time, service times and depot load/unload time) is limited [17]. Depending on version, capacity constraints can be applied to the whole vehicle route or only to a route section between subsequent depots (if we allow for vehicle's multiple comebacks to depots, where it is unloaded). In the Time-dependent Vehicle Routing Problem (TDVRP) travel times between clients depend on the moment of travel start. This problem version can model traffic in road networks (i.e. lower speeds in peak hours) [14]. In stochastic versions of VRP clients demands or travel times are not known beforehand (they are random variables) [21].

In the article we deal mostly with VRP variants including capacity, drivers work time and time-windows constraints: they can be classified as DCVRPTW (Distance Constrained Capacitated Vehicle Routing Problems with Time Windows). Both strict time-windows as well as soft time-windows versions are implemented in our algorithm.

3. Problem formulation as mixed-programming problem

The definition of DCVRPTW with strict time windows we use here is based on the flex version of DCVRP [17] (multiple comebacks to depots are allowed) but with additional time-windows constraints added. Let K be the set of vehicles. Let N be the set of customers, H_0 the set of original depots, $H_1...H_{B-1}$ sets of intermediate depots (which vehicles can come back to inside their routes), H_B the set of final depots and V the set of all nodes (the union of all previously defined sets). Let d_i be nonnegative demand of customer $i \in N$. Let c_{ij} be the cost of travelling from node i to node j and t_{ij} its travel time. Let s_i be service time of customer i . Let o_i be time-window opening of customer i and cl_i its closing time. Let Q be the maximal vehicle load while R be the maximal vehicle route time. Let x_{ijk} be a binary variable indicating if vehicle k travels from node i to node j . Let e_{ik} be an auxiliary, binary variable indicating if customer i is visited by vehicle k . It is defined as $\sum_{j \in V} x_{ijk}$. Let y_k be a binary variable indicating if vehicle k is used in the solution. Let z_{ik} be the total load served by vehicle k between its last visit to depot and visit to node i (including the load of node i). It is set to 0 at depot nodes. Let w_{ik} be the arrival time of vehicle k at node i . It is initialized to 0 at original depot nodes (that start vehicle's route).

The goal is to minimize formula (1)

$$\sum_{i \in V} \sum_{j \in V \setminus \{i\}} \sum_{k \in K} x_{ijk} c_{ij} + \sum_{k \in K} y_k m \quad (1)$$

subject to constraints:

$$\sum_{j \in V \setminus \{i\}} \sum_{k \in K} x_{ijk} = 1 \quad \forall i \in N \quad (2)$$

$$\sum_{i \in V \setminus \{j\}} x_{ijk} - \sum_{i \in V \setminus \{j\}} x_{jik} = 0 \quad \forall j \in N, k \in K \quad (3)$$

$$\sum_{i \in H_0} \sum_{j \in N} x_{ijk} - y_k = 0 \quad \forall k \in K \quad (4)$$

$$\sum_{i \in H_{b-1}} \sum_{j \in N} x_{ijk} - \sum_{i \in H_b} \sum_{j \in N} x_{ijk} = 0 \quad \forall k \in K, H_b \in \{H_1, \dots, H_B\} \quad (5)$$

$$\sum_{i \in N} x_{ijk} - \sum_{i \in N} x_{jik} \geq 0 \quad \forall j \in \{H_1, \dots, H_{B-1}\}, k \in K \quad (6)$$

$$x_{ijk} = 0 \quad \forall i \in H_B, j \in N, k \in K \quad (7)$$

$$(z_{ik} + d_j - z_{jk}) \leq M(1 - x_{ijk}) \quad \forall i \in V, j \in N \setminus \{i\}, k \in K \quad (8)$$

$$(z_{ik} + d_j - z_{jk}) \geq -M(1 - x_{ijk}) \quad \forall i \in V, j \in N \setminus \{i\}, k \in K \quad (9)$$

$$(w_{ik} + s_i + t_{ij} - w_{jk}) \leq M(1 - x_{ijk}) \quad \forall i \in V, j \in \{N, H_1, \dots, H_B\} \setminus \{i\}, k \in K \quad (10)$$

$$(w_{ik} + s_i + t_{ij} - w_{jk}) \geq -M(1 - x_{ijk}) \quad \forall i \in V, j \in \{N, H_1, \dots, H_B\} \setminus \{i\}, k \in K \quad (11)$$

$$0 \leq z_{ik} \leq Q \quad \forall i \in N, k \in K \quad (12)$$

$$0 \leq w_{ik} \leq R \quad \forall i \in \{N, H_1, \dots, H_B\}, k \in K \quad (13)$$

$$e_{ik} \cdot o_i \leq e_{ik} \cdot w_{ik} \leq e_{ik} \cdot cl_i \quad \forall i \in N, k \in K \quad (14)$$

$$z_{ik} = 0 \quad \forall i \in \{H_0, \dots, H_B\}, k \in K \quad (15)$$

$$w_{ik} = 0 \quad \forall i \in H_0, k \in K \quad (16)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i, j \in V, k \in K, i \neq j \quad (17)$$

$$y_k \in \{0, 1\} \quad \forall k \in K \quad (18)$$

In Equation (1) constant m set to 0 means that only total routes cost is minimized while setting it to a sufficiently large value will minimize number of cars used and then total cost. Constraint (2) means that each customer must be visited exactly once by exactly one vehicle. Equation (3) is a flow conversion constraint (vehicle arriving at a given customer node has to leave it afterwards). Equation (4) is a relation between variables x and y . Equations (5) and (6) are conversion of flow through depots constraints. Constraint (7) eliminates any flow outcoming from final depots. Constraints (8) and (9) define relations between load variables and customer demands.

Constraints (10) and (11) define relations between time variables, travel times and customers service times. In those constraints we assume sufficiently large constant M (i.e. larger than the sum of client demands in Equations (8) and (9)). Constraint (12) represents the load limits and constraint (13) the route time limits. Constraint (14) forces time window obedience but only for customers visited by vehicle k (therefore e variable is introduced in the formula). Constraints (15) and (16) reset load and time variables in depots (in all depots in case of load and in starting depots in case of time). Constraint (17) and (18) forces variables x and y to be binary.

In DCVRPTW with soft time windows we allow vehicles to arrive after customers closing time. Equation (14) from the above definition is replaced with

$$e_{ik} \cdot o_i \leq e_{ik} \cdot w_{ik} \quad \forall_{i \in N, k \in K} \quad (19)$$

In this variant we also try to minimize the total time-windows delay of all visited customers. A delay is non-zero for a customer visited after its TW close time and is defined as the difference between vehicle arrival time and TW close time. Therefore it is a bi-objective optimization with two optimization criteria (for total cost and total time windows delay):

$$\sum_{i \in V} \sum_{j \in V \setminus \{i\}} \sum_{k \in K} x_{ijk} c_{ij} + \sum_{k \in K} y_k m \quad (20)$$

$$\sum_{i \in N, k \in K} e_{ik} \cdot \max(0, w_{ik} - c_i) \quad (21)$$

4. State of the art

The VRP is an NP-hard optimization problem [19] and no polynomial-time exact algorithms are known. Branch-and-cut solutions [3] are among most effective exact approaches for problems from TSP and VRP family. However, for large problem instances their computation time is usually impractically long. Therefore the main focus of researchers has been on heuristics and metaheuristics, which can find satisfactory solutions (but usually not optimal) in shorter execution time. One of the first heuristic for VRP was savings algorithm [8], where initially each route includes only one customer and routes are merged according to a criterion maximizing distance reduction until no further merges are feasible regarding the problem constraints. Metaheuristics have been successfully applied to solve many optimization problems and they include exploration (discovering new regions in solutions space) and exploitation (searching promising regions intensely) phases. Some of them operate on single solutions (i.e. iterated local search (ILS) [2] or tabu search (TS) [13]) and some of them maintain a population of solutions: i.e. nature-inspired methods like evolutionary algorithms (EA) [25], particle swarm optimization (PSO) [1] and ACO metaheuristics. Some researchers also tried the combination of state-of-the art metaheuristics with machine learning [11, 26].

5. Algorithm description

The authors proposed the ACO metaheuristic [12] to solve routing problems. This approach is inspired by the behaviour of ants and is well suited to solve graph optimization problems. It is a probabilistic, multi-agent approach, which belongs to swarm intelligence methods. Each ant (an agent) moves between various states and constructs a solution. The probability of a transition between states depends on *a priori* knowledge (desirability) and on pheromone levels (swarm intelligence – knowledge gathered during the algorithm search). The procedure of solutions construction is repeated multiple times allowing the colony to learn more about the problem instance. In graph optimization problems each ant traverses edges until a feasible solution is constructed, desirability is directly associated with edge costs (i.e. travel time or distance) while edges pheromone levels are updated according to the quality of solutions they form. Due to the fact that each ant constructs its solution independently ACO parallelization can be efficiently implemented [20, 28], which is also an important quality of this metaheuristic. In addition, the specificity of ACO solution construction makes adaptation to various optimization targets and creation a flexible tool solving real-life problems easier. The algorithm pseudocode is shown at the end of this section.

5.1. Solution construction

The algorithm consists of n main iterations. In every iteration each of k ants builds a solution beginning from a given starting vertex. To preserve solutions feasibility (regarding constraints like max. work time, time-windows or car capacity) ants can only choose edges that do not cause constraint violations. The probability of choosing edge ij is given by the following formula:

$$p_{ij} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{ik \in feasibleMoves} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}} \quad (22)$$

In the above formula η_{ij} represents desirability of a given edge, which is inversely proportional to its cost (distance or travel time, depending on optimization criteria), τ_{ij} represents colony knowledge regarding a given edge, while α and β parameters control the influences of colony knowledge and desirability knowledge.

If a solution being built is still not complete and no new vertex can be visited without violating constraints then an ant goes back to the depot vertex and a new subpath is constructed (in a new work time slot, if necessary).

5.2. Local optimization

After a feasible solution is build it undergoes a series of local optimizations. ACO performs an exploration of solution space and finds promising regions while local search methods enable for a better exploitation of those regions. Local search operators include 2-opt, insert and delete methods.

5.2.1. 2-opt

Two-opt is an operator that chooses two non-adjacent edges in a path: $(p_i p_{i+1})$ and $p_j p_{j+1}$ and replaces them with edges $(p_i p_j)$ and $(p_{i+1} p_{j+1})$. In other words, it inverts a given path fragment $(p_{i+1} \dots p_j)$. In our solution we apply a hill climbing version, which modifies a path as long as there exists an improving move (regarding our optimization criterion or criteria). The operator is presented in Figure 1.

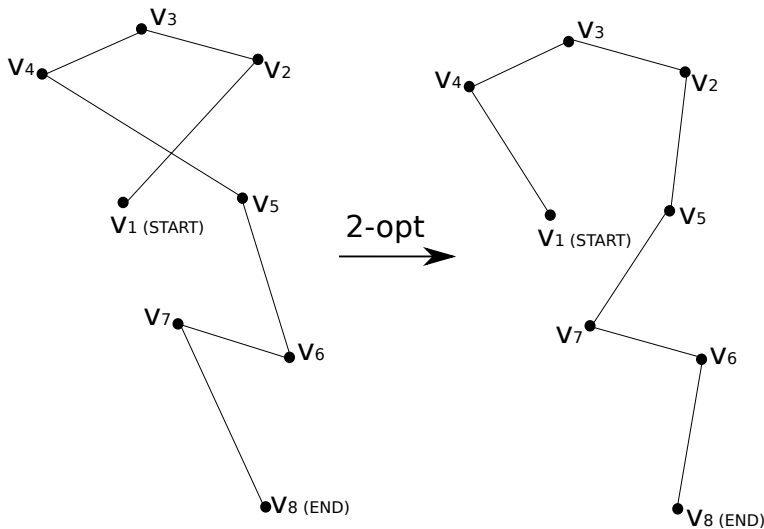


Figure 1. An example of 2-opt local search. First move inverts path fragment v_2 - v_3 - v_4 and the second move inverts fragment v_6 - v_7 . As a result a local optimum is reached

5.2.2. Insert and delete

Another local search procedure used involves insert and delete operators. After deleting a given group of vertices (they can be chosen randomly or according to a heuristic) they are inserted back into a solution in such places that the solution cost increase is minimized. This procedure can be applied multiply for varying number of modified vertices until no further improvement is found.

5.2.3. Generalized move – path look improvement and final optimization

The best final paths undergo an additional local search, which moves groups of points between paths in order to maximize a certain criterion. In commercial applications, besides meeting optimization goals, a visual aspect of solutions can be important (how clients perceive visualization of solutions). For example it is visually better if one route serves mostly a certain area/city district without serving too many clients from other districts. Normally even high-quality solutions do not guarantee this as clients from various districts are often on vehicles' way to other areas. This operator

can improve compactness of paths without deteriorating solutions quality. It is based on mean square error (MSE) measure known from clustering.

Alternatively, this operator can serve as the last optimization of the resulting solution (improving distance instead of MSE) if the only goal is to optimize path cost.

5.3. Pheromones update

After solutions are constructed and optimized it is time for pheromones update. Initially, all pheromones undergo the procedure of evaporation (their value decrease at the end of an iteration). Afterwards, the best solutions (from the current iteration and from the whole algorithm run) are used and the edges they consist of have pheromone levels increased. In this way the colony updates its knowledge regarding the solved problem instance. The update procedure is given by the following formula:

$$\tau_{ij} \leftarrow \tau_{ij} \cdot (1 - \rho) + l \cdot \gamma \quad (23)$$

In the formula ρ is an evaporation coefficient, γ is an update coefficient and l is the number of solutions (among the best ones) including edge ij . Parameters ρ and γ control the rate of colony knowledge change. The bigger those values the faster knowledge update takes place. Pheromone amounts on edges are limited to an interval $[\tau_{min}, \tau_{max}]$ as it is in the min-max ant system (MMAS).

5.4. Construction of pareto set approximations

In some problem versions, more than one optimization criteria is applied. Each solution can be then described by a criterion vector $(y_1, y_2, \dots, y_n) \in R^n$. Let $y' \succ y''$ mean that solution y' dominates solution y'' . The definition of domination is as follows:

$$y' \succ y'' \iff \forall_{1 \leq i \leq n} (y'_i \leq y''_i) \wedge \exists_{1 \leq i \leq n} (y'_i < y''_i) \quad (24)$$

This means that y' is not worse than y'' in any optimization criterion but is better in at least one (assuming a minimization problem). A set of all non-dominated solutions is called pareto front. The algorithm maintains a set of pairwise non-dominated solutions (approximation of the pareto front). Let Y be the set of all feasible solutions (their optimization criteria results). Pareto front P definition is given below:

$$P = \{y \in Y : \neg \exists_{y' \in Y} (y' \succ y)\} \quad (25)$$

In case of multiple criteria optimization each ant has pseudorandom weights assigned (from $[0, 1]$ interval), which signal the importance of each optimization target. Then pheromones and desirability computations are based on those weights and optimization targets properties. Let (η_1, \dots, η_n) and (τ_1, \dots, τ_n) represent vectors of desirability and colony knowledge for all optimization targets and for one particular move and let (w_1, \dots, w_n) be a vector of optimization target weights of a given

ant. Then composite desirability and colony knowledge for all optimization targets is calculated as follows:

$$\tau = \prod_{1 \leq i \leq n} (\tau_i \cdot w_i) \quad (26)$$

$$\eta = \prod_{1 \leq i \leq n} (\eta_i \cdot w_i) \quad (27)$$

Pseudorandom distribution of weights among ants lets the algorithm to approximate various fragments of the pareto front.

Algorithm 1 ACO

```

1: bestSolutions =  $\phi$ 
2: for  $i \leftarrow 1 \dots n$  do
3:   solutions =  $\phi$ 
4:   for  $j \leftarrow 1 \dots k$  do
5:     currentSolution = (startVertex)
6:     while currentSolution is not complete and there are feasible moves do
7:       choose a vertex  $w$  (probability based on desirability and pheromones)
8:       append vertex  $w$  to currentSolution
9:     end while
10:    optimize locally currentSolution
11:    add currentSolution to solutions
12:  end for
13:  update bestSolutions with solutions
14:  update pheromones based with top solutions from solutions and bestSolutions
15: end for
16: return bestSolutions

```

6. Experiments

The algorithm was applied to solve practical optimization problems for a few transportation companies. Our tool is flexible and enables us to solve various versions of VRP family (regarding constraints and optimization targets) and therefore it is well suited to meet specific needs of the companies. Experiments were conducted on a computer with 3.5 GHz processor (4 cores), 16 GB of RAM and Linux operation system. The application was written in Scala programming language. The algorithm's parameters values are given in Table 1. Real distances and approximate travel times between client addresses were obtained from Open Route Service [22]. For comparison purposes we decided to use OR optimization tool [23] for some companies test instances. We compared our method with Greedy Descent (GD) and Guided Local Search (GLS) metaheuristics [31]. The metaheuristics time limit was set to the same values as ACO execution times. In addition, we executed our algorithm on VRPTW benchmarks with known optimal solutions (Solomon instances) and compared with

other metaheuristics. The results presented are average of multiple runs (to obtain some statistics) but usually in our web application one run is performed (it can be reconfigured to choose the best of n runs too).

Table 1
ACO parameters values

Parameter	Name	Value
n	Iterations count	150
k	Ants count	25
α	Colony knowledge coefficient	20
β	Desirability coefficient	40
ρ	Evaporation coefficient	0.01
γ	Pheromone update coefficient	0.03

6.1. Parameters tuning and analysis

Iterations count and ants count were set in a way to reach high-quality solutions in reasonable execution times. For a given ants number iterations count was determined empirically. In Table 2 there is a comparison of algorithm results for various ants and iterations count. Due to increase of exploration power ACO generates better results with ants count increase. We chose 25 ants as for larger count the increase of solution quality is getting smaller while execution time grows significantly. In Figure 2 an exemplary algorithm run is presented (iterations vs best result). The result before final local optimization (in post-ACO phase) is given.

Table 2
Tuning results for ants count – results for catering company network (Gdansk)

Ants	Iterations	Result	Gap [%]	Execution time
10	75	440.8	2.8	19
15	100	436.2	1.7	35
25	150	428.9	–	78
40	250	426.7	–0.5	207

To finetune the remaining parameter values of the ACO we ran the algorithm with four possible values of α (10, 20, 40, 80), four values of β (10, 20, 40, 80), three values of ρ (0.01, 0.03 and 0.1) and three values of γ (0.01, 0.03 and 0.1). Therefore, 144 combinations of parameter values were tested on two instances (catering company instance – Gdansk and one of Solomon VRPTW benchmark instances). For each combination in 4-dimensional space of parameter values ACO was run 10 times for both test instances and we computed average gap (to the optimal solution in case of Solomon instance and to the best known solution in case of Gdansk instance). In this way we determined the best set of parameter values.

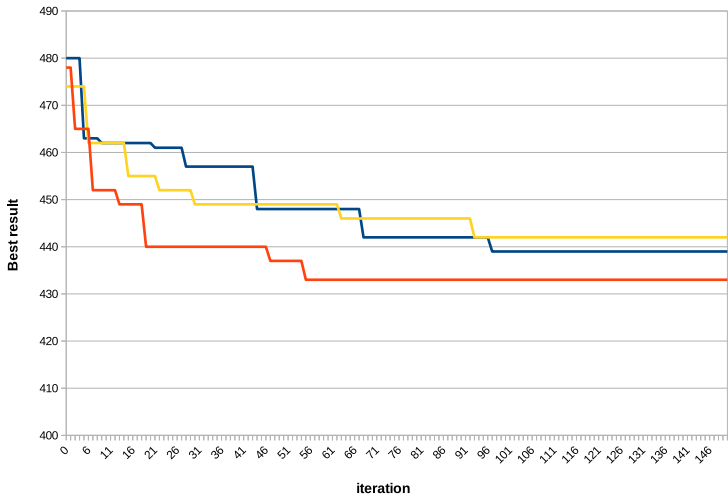


Figure 2. Exemplary algorithm runs: best result so far vs iteration number, Gdansk network

In Table 3 we present averaged tuning results for slow evaporation and moderate pheromone update rate configuration. It can be seen that configurations with the smallest utilization of the colony knowledge ($\alpha = 10$) produced the worst result while there are minor differences between configurations varying in desirability coefficient. The best configuration has moderate values of desirability and pheromone coefficients and most of them are within 0.5 percent of the best one. Small differences indicate the algorithm’s robustness regarding parameter values change and can be also associated with strong influence of local search procedures on the final result (they exploit solution space areas searched by the metaheuristic).

Table 3

Results of parameter tuning for slow evaporation ($\rho = 0.01$) and moderate pheromone update rate ($\gamma = 0.03$). Average results for given α and β values as well as the best configuration are in bold. Combined average gap of both tuning test instances is presented

α/β	10	20	40	80	avg.
10	2.5	2.8	2.0	1.9	2.3
20	1.2	1.5	1.0	1.6	1.3
40	1.2	1.3	1.4	1.5	1.3
80	1.2	1.4	1.3	1.7	1.4
avg.	1.5	1.8	1.4	1.7	–

In Table 4 tuning results for moderate evaporation and fast pheromone update rate are presented. Generally, differences between various α and β value sets are smaller than in the previous configuration. With faster pheromone evaporation and update even configurations with smaller colony knowledge utilization ($\alpha = 10$) have reached high-quality solutions.

Table 4

Results of parameter tuning for moderate evaporation ($\rho = 0.03$) and fast pheromone update rate ($\gamma = 0.1$). Average results for given α and β values as well as the best configuration are in bold. Combined average gap of both tuning test instances is presented

α/β	10	20	40	80	avg.
10	2.0	1.2	1.3	2.0	1.6
20	1.4	1.2	1.4	1.6	1.4
40	1.9	1.5	1.2	1.8	1.6
80	1.4	1.3	1.7	2.3	1.7
avg.	1.6	1.3	1.4	1.8	–

In Table 5 there are tuning results for high evaporation and moderate pheromone update rates. This time the best results are reached for smaller α values. It seems that increasing evaporation and update rates compensates for a lower pheromone trail coefficient. On the other hand, the algorithm consistently generates worse results for the highest value of desirability coefficient ($\beta = 80$). This can be associated with extensive local search methods, which already utilize the problem-specific knowledge so there is no need to overuse the heuristic component during paths construction.

Table 5

Results of parameter tuning for high evaporation ($\rho = 0.1$) and moderate pheromone update rate ($\gamma = 0.03$). Average results for given α and β values as well as the best configuration are in bold. Combined average gap of both tuning test instances is presented

α/β	10	20	40	80	avg.
10	2.0	1.2	1.3	2.0	1.6
20	1.8	1.7	1.9	2.0	1.8
40	1.6	1.9	1.9	2.1	1.9
80	1.8	1.9	2.6	3.1	2.3
avg.	1.8	1.7	1.9	2.3	–

In Table 6 tuning results for various pheromone evaporation and update rates are presented (global average). It can be seen that the best results are generated when update rate is three times higher than evaporation rate. The differences are small but higher values of update rates generally are associated with better average results.

Table 6

Tuning results for various values of pheromone evaporation and update rates averaged over all values of α and β . The best configurations in bold

ρ/γ	0.01	0.03	0.1
0.01	2.0	1.6	1.8
0.03	2.0	2.0	1.5
0.1	1.8	1.9	1.9

6.2. Catering company results

The algorithm was applied to help a Polish catering company with their route planning in Polish cities and regions (Warsaw, Gdansk and north-eastern Poland – see Fig. 3). The transport of goods was scheduled for one night. Clients time windows (of sizes: 4–11 hours) were the main constraint on vehicles' routes size. No time limit on routes exists in this instance. There were 1539 clients in Warsaw and 16 vehicles available. In NE Poland there were 2117 clients and 29 vehicles available. In Gdansk there were 345 clients and 4 vehicles available. The goal of the optimization was to minimize the total distance covered by all vehicles without violating time-windows constraints. We tested networks for two versions of the algorithm: a commercial variant (improving paths look as final optimization) and performance variant (further distance improvement in the final optimization). The results are given in Table 7. It can be seen that the version which incorporates further distance improvement is on average 4–5 percent better than the commercial version. In Table 8 there is a comparison of ACO with metaheuristics from the OR tool. ACO clearly outperforms other metaheuristics for Warsaw and NE Poland data sets. Average advantage over GLS is 3.2 percent and over GD 4.8 percent.

Table 7

Catering company results. Results are the average of 30 algorithm runs. The version with path look improvement is marked as ACO.P while the version with final distance improvement is marked as ACO.D. Solution distances are given in kilometers while execution times in seconds

Instance	ACO.P			ACO.D		
	Vehicles used	Distance	Execution time	Vehicles used	Distance	Execution time
Warsaw	12	1231.7	108	12	1211.3	131
Gdansk	4	437.4	70	4	428.1	78
NE Poland	28	7388.7	650	28	7180.4	755

Table 8

Comparison of ACO with GD and GLS (catering company results). Results are the average of 30 runs. Confidence intervals (alfa = 0.05) given in +/- column

Instance	ACO.D		GD			GLS		
	Distance	+/-	Distance	+/-	Gap [%]	Distance	+/-	Gap [%]
Gdansk	428.1	1.0	460.3	3.9	7.5	442.8	3.7	3.4
Warsaw	1211.3	5.7	1210.8	4.3	0.0	1223.3	5.1	1.0
NE Poland	7180.4	35.1	7675.4	70.6	6.9	7558.2	63.2	5.2
Average	–	–	–	–	4.8	–	–	3.2

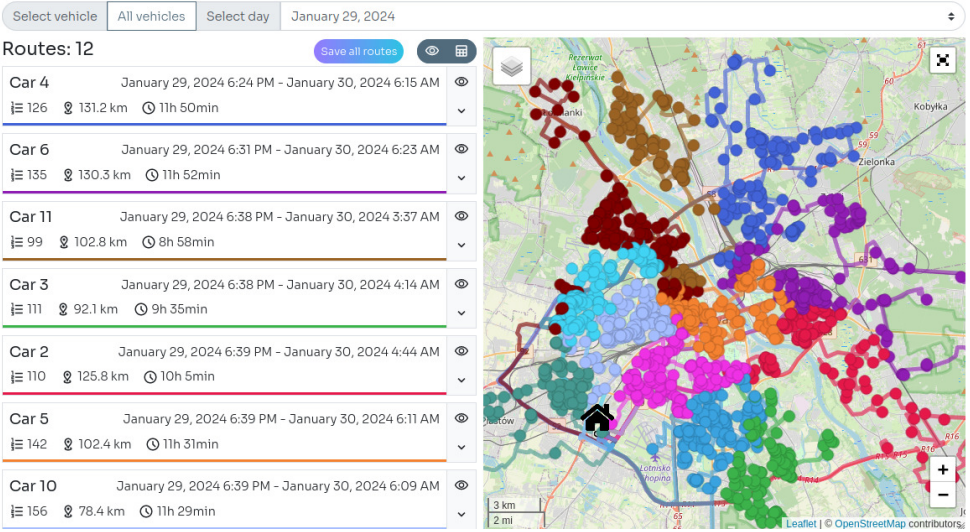


Figure 3. Algorithm run results for the catering company displayed in the application window (for the city of Warsaw)

6.3. Pharmaceutical wholesaler company results

The algorithm was applied to generate solutions for a pharmaceutical company (see Fig. 5). In one version a depot was located in Krakow (301 client addresses to visit and 15 vehicles) and in the other version in Rzeszow (131 client addresses to visit and 8 vehicles). There were two distinct capacity limits: one was a standard truck capacity and the other was a maximal capacity for cold products (that have to be transported in low temperatures). Drivers work hours were 6–18 (12 hour route time limit) for 5 days a week but some of them did not have a fixed limit. Therefore, this is a generalization of DCVRPTW, where drivers work hours replace standard route time limit. In this version there are soft time windows and one of the objectives was to minimize total violation of time windows (some clients were served later than its time-window close time). Therefore we have two optimization targets (total time windows delay and total distance). In this configuration, depending on clients flexibility, one can choose any of the generated solutions in our application (clients TWs were usually narrow and mostly in the first half of the day).

In Figure 4 the results of ACO bi-objective optimization are compared with GLS and GD algorithms. To generate various solutions in the OR tool (which has one optimization target) a few values of penalties (for exceeding soft time window upper bound) were used and each configuration was executed a few times. ACO outperformed other metaheuristics regarding total TW delay reaching the lowest values on pareto front approximations. All three metaheuristics were very similar regarding total distance (GD had on average 1 percent longer routes than GLS and ACO).

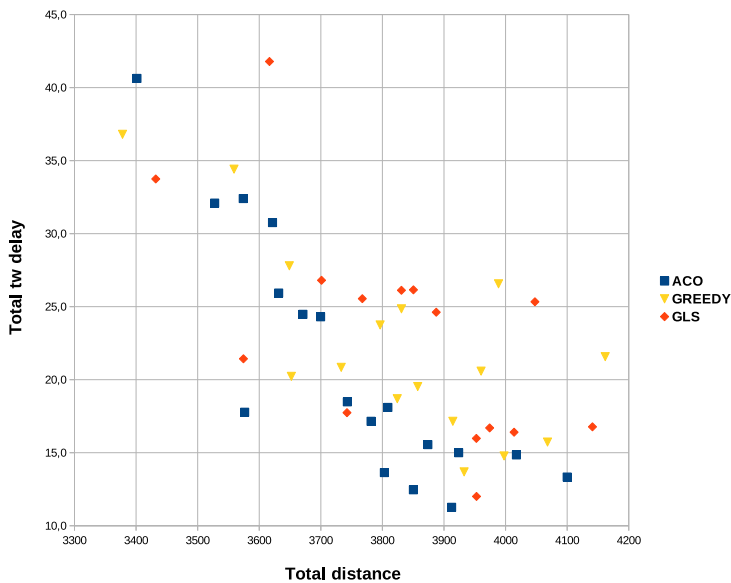


Figure 4. Comparison of ACO, GLS and GD metaheuristics results, bi-objective optimization for the pharmaceutical company (depot in Krakow)

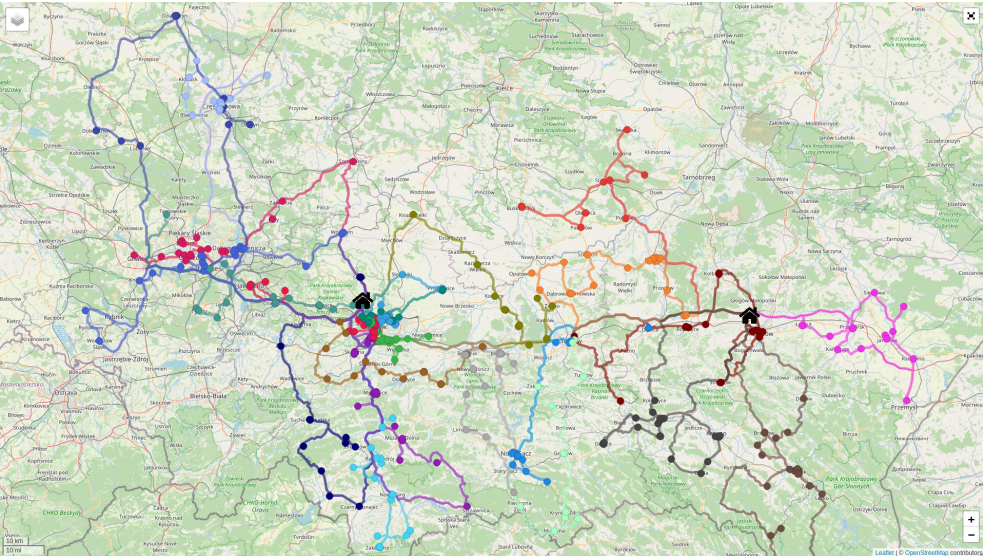


Figure 5. Algorithm run results for the pharmaceutical company (Krakow and Rzeszow combined results)

6.4. Delivery company results

We also planned routes for a delivery company (see Fig. 6). The goal was to plan delivery in Warsaw and surrounding towns. There were 5090 clients and 116 available vehicles. Deliveries were planned for one day and the work time limit for every vehicle was 10 hours. It was a multi-objective optimization as there were two optimization targets: total distance and total work time. The best results by each objective are presented in Table 9. The approximate pareto-front is not wide due to both criteria results being correlated with each other.

Table 9
Delivery company results. Results are the average of 10 algorithm runs

Criterion	Vehicles used	Total distance [km]	Total work time
Best by distance	104	10940.1	1005 h 35 min
Best by total work time	104	11130.2	1001 h 45 min

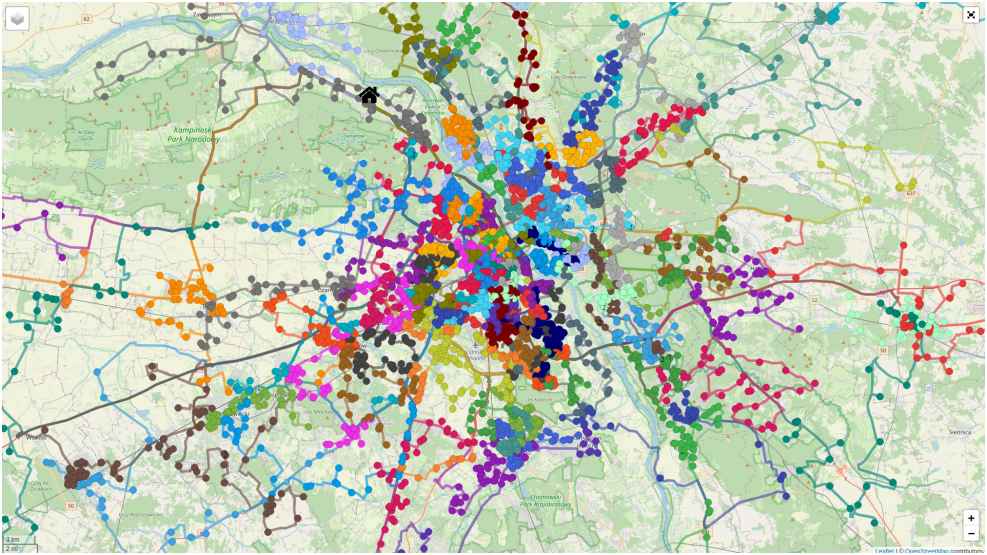


Figure 6. Algorithm run results for the delivery company (Warsaw and surroundings)

6.5. Benchmark instances results

To compare the solutions of our algorithm to known optimal solutions we executed it on some VRPTW benchmarks (Solomon C instances [9]). The results are presented in Table 10. It can be seen that the algorithm reaches optimal or nearly optimal results in short execution times. Only for two test instances the algorithm generated non-optimal solutions.

In Table 11 a comparison of ACO and other metaheuristics is presented. The compared metaheuristics are local search (LS) [6], tabu search (TS) [18], evolutionary algorithm (EVO) [15] and hybrid of TS and simulated annealing (HYB) [29]. Their results are among the best achieved by heuristic solutions for VRPTW instances. It can be seen that ACO produced the best results on average.

Table 10

Solomon C instances results. The average of 30 algorithm runs, standard deviations and gaps to optimal solutions are presented. Execution time is given in seconds

Instance	Vertex count	Vehicles count	Maximum capacity	Distance	Standard deviation	Gap [%]	Vehicles used	Execution time
C101	101	25	200	827.3	0.0	0.0	10	4.9
C102	101	25	200	827.3	0.0	0.0	10	2.1
C103	101	25	200	826.3	0.0	0.0	10	2.2
C104	101	25	200	828.7	2.9	0.7	10	2.1
C105	101	25	200	827.3	0.0	0.0	10	2.1
C106	101	25	200	827.3	0.0	0.0	10	1.9
C107	101	25	200	827.3	0.0	0.0	10	1.7
C108	101	25	200	827.3	0.0	0.0	10	2.0
C109	101	25	200	827.3	0.0	0.0	10	2.2
C201	101	25	700	589.1	0.0	0.0	3	1.8
C202	101	25	700	589.1	0.0	0.0	3	2.2
C203	101	25	700	588.7	0.0	0.0	3	2.4
C204	101	25	700	593.2	2.6	0.9	3	2.4
C205	101	25	700	586.4	0.0	0.0	3	2.0
C206	101	25	700	586.0	0.0	0.0	3	2.1
C207	101	25	700	585.8	0.0	0.0	3	2.0
C208	101	25	700	585.8	0.0	0.0	3	1.8

Table 11

Average results for Solomon C instances – comparison of the ACO and other metaheuristics

Algorithm	C1 instances	C2 instances
ACO	827.3	588.1
LS	832.9	593.5
TS	832.1	589.9
EVO	828.4	589.8
HYB	841.9	612.4

7. Conclusion

In the paper we presented the ant-colony optimization metaheuristic for the Vehicle Routing Problems family. The algorithm can be applied to solve various variants of VRP and is a part of a web application, which meets practical needs of transport

industry companies. The algorithm obtained satisfactory solutions in acceptable execution times and had an advantage over other compared metaheuristics. The selection of the metaheuristic and local optimization methods provides good solution quality to execution time ratio. It is crucial for real-life applications, because quality gives operational savings for users, but the computation time is in some cases restricted by real-life operations and the time slot between the moment of collecting the last orders to be planned and the time when the drivers need to know their routes to prepare to departure.

Acknowledgements

Our work was done as a part of project “TRASA – development and validation of algorithms for routes optimization and resources allocation” [30] and was financed by Intelligent Development Operational Program 2014–2020, sub-program: Industrial Research and Development Projects Carried out by Enterprises.

References

- [1] Ai J., Kachitvichyanukul V.: A Particle Swarm Optimisation for Vehicle Routing Problem with Time Windows, *International Journal of Operational Research*, vol. 6(4), pp. 519–537, 2009. doi: 10.1504/ijor.2009.027156.
- [2] Álvarez A., Munari P.: Metaheuristic approaches for the vehicle routing problem with time windows and multiple deliverymen, *Gestão & Produção*, vol. 23(2), pp. 279–293, 2016.
- [3] Araque J.R., Kudva G., Morin T.L., Pekny J.F.: A branch-and-cut algorithm for vehicle routing problems, *Annals of Operations Research*, vol. 50, pp. 37–59, 1994. doi: 10.1007/bf02085634.
- [4] Balakrishnan N.: Simple heuristics for the vehicle routing problem with soft time windows, *The Journal of the Operational Research Society*, vol. 44(3), pp. 279–287, 1993. doi: 10.1038/sj/jors/0440308.
- [5] Bellmore M., Nemhauser G.L.: The Traveling Salesman Problem: A Survey, *Operation Research*, vol. 16, pp. 538–558, 1986.
- [6] Bräysy O.: Fast local searches for the vehicle routing problem with time windows, *Information Systems and Operations Research*, vol. 41, pp. 319–330, 2003.
- [7] Bräysy O., Gendreau M.: Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms, *Transportation Science*, vol. 39(1), pp. 104–118, 2005. doi: 10.1287/trsc.1030.0056.
- [8] Clarke G.U., Wright J.W.: Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research*, vol. 12(4), pp. 568–581, 1964. <http://www.jstor.org/stable/167703>.
- [9] *CVRPLIB Capacitated Vehicle Routing Problem Library*. Online: <http://vrp.galgos.inf.puc-rio.br/index.php/en/>.
- [10] Dantzig G.B., Ramser J.H.: The truck dispatching problem, *Management Science*, vol. 6, pp. 80–91, 1959. doi: 10.1287/mnsc.6.1.80.

- [11] Di Puglia Pugliese L., Ferone D., Festa P., Guerriero F., Macrina G.: Combining variable neighborhood search and machine learning to solve the vehicle routing problem with crowd-shipping, *Optimization Letters*, vol. 17, pp. 1981–2003, 2023. doi: 10.1007/s11590-021-01833-x.
- [12] Dorigo M., Stützle T.: *Ant Colony Optimization*, The MIT Press, Cambridge, 2004.
- [13] Gendreau M., Hertz A., Laporte G.: A Tabu Search Heuristic for the Vehicle Routing Problem, *Management Science*, vol. 40(10), pp. 1207–1393, 1994. doi: 10.1287/mnsc.40.10.1276.
- [14] Gmira M., Gendreau M., Lodi A., Potvin J.Y.: Tabu search for the time-dependent vehicle routing problem with time windows on a road network, *European Journal of Operational Research*, vol. 288, pp. 129–140, 2021. doi: 10.1016/j.ejor.2020.05.041.
- [15] Homberger J., Gehring H.: A two-phase hybrid metaheuristic for the vehicle routing problem with time windows, *European Journal of Operations Research*, vol. 62, pp. 220–238, 2005. doi: 10.1016/j.ejor.2004.01.027.
- [16] Ibrahim A.A., Lo N., Abdulaziz R.O., Ishaya J.A.: Capacitated Vehicle Routing Problem, *International Journal of Research – Granthaalayah*, vol. 7(3), pp. 310–327, 2019. doi: 10.29121/granthaalayah.v7.i3.2019.976.
- [17] Kek A.G.H., Cheu R.L., Meng Q.: Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots, *Mathematical and Computer Modelling*, vol. 47(1–2), pp. 140–152, 2008. doi: 10.1016/j.mcm.2007.02.007.
- [18] Lau H.C., Sim M., Teo K.M.: Vehicle routing problem with time windows and a limited number of vehicles, *European Journal of Operational Research*, vol. 148, pp. 559–569, 2003. doi: 10.1016/s0377-2217(02)00363-6.
- [19] Lenstra J.K., Rinnooy Kan A.H.G.: Computational Complexity of Discrete Optimization Problems, *Annals of Discrete Mathematics*, vol. 4, pp. 121–140, 1979. doi: 10.1016/s0167-5060(08)70821-5.
- [20] de Melo Menezes B.A., Herrmann N., Kuchen H., de Lima Neto F.B.: High-Level Parallel Ant Colony Optimization with Algorithmic Skeletons, *International Journal of Parallel Programming*, vol. 49, pp. 776–801, 2021. doi: 10.1007/s10766-021-00714-1.
- [21] Niu Y., Shao J., Xiao J., Song W., Cao Z.: Multi-objective evolutionary algorithm based on RBF network for solving the stochastic vehicle routing problem, *Information Sciences*, vol. 609, pp. 387–410, 2022. doi: 10.1016/j.ins.2022.07.087.
- [22] *Openroute Service*. Online: <https://openrouteservice.org/>.
- [23] *OR-Tools*. Online: <https://developers.google.com/optimization>.
- [24] Ostrowski K., Karbowska-Chilinska J., Koszelew J., Zabielski P.: Evolution-inspired local improvement algorithm solving orienteering problem, *Annals of Operations Research*, vol. 253, pp. 519–543, 2017. doi: 10.1007/s10479-016-2278-1.

- [25] Prins C.: A simple and effective evolutionary algorithm for the vehicle routing problem, *Computers and Operations Research*, vol. 31(12), pp. 1985–2002, 2004. doi: 10.1016/s0305-0548(03)00158-8.
- [26] Qi R., Li J.Q., Wang J., Jin H., Han Y.Y.: QMOEA: A Q-learning-based multi-objective evolutionary algorithm for solving time-dependent green vehicle routing problems with time windows, *Information Sciences*, vol. 608, pp. 178–201, 2022. doi: 10.1016/j.ins.2022.06.056.
- [27] Starzec M., Starzec G., Byrski A., Turek W.: Distributed ant colony optimization based on actor model, *Parallel Computing*, vol. 90(1), 102573, 2019. doi: 10.1016/j.parco.2019.102573.
- [28] Stützle T.: Parallelization strategies for Ant Colony Optimization, *Lecture Notes in Computer Science*, vol. 1498, pp. 722–731, 1998.
- [29] Tan K.C., Lee K.O.: Artificial intelligence heuristics in solving vehicle routing problems with time window constraints, *The Engineering Applications of Artificial Intelligence*, vol. 14, pp. 825–837, 2001. doi: 10.1016/s0952-1976(02)00011-8.
- [30] *TRASA – development and validation of algorithms for routes optimization and resources allocation*. Online: <https://getsent.io/en/projects/trasa>.
- [31] Voudouris C., Tsang E.P., Alsheddy A.: Guided Local Search. In: M. Gendreau, J.Y. Potvin (eds.), *Handbook of Metaheuristics*, pp. 321–361, Springer, 2010.

Affiliations

Krzysztof Ostrowski

Białystok University of Technology, Faculty of Computer Science, ul. Wiejska 45A,
15-351 Białystok; Sentio sp. z o.o., ul. Warszawska 6/32, 15-063 Białystok,
k.ostrowski@pb.edu.pl

Mateusz Starzec

Sentio sp. z o.o., ul. Warszawska 6/32, 15-063 Białystok, m.starzec@getsent.io

Grażyna Starzec

AGH University, Faculty of Computer Science, al. Adama Mickiewicza 30, 30-059 Kraków;
Sentio sp. z o.o., ul. Warszawska 6/32, 15-063 Białystok, g.starzec@getsent.io

Received: 13.06.2024

Revised: 15.06.2024

Accepted: 15.06.2024

SAIB BOUTHINA
MOHAMED-RIDA ABDESSEMED
RIADH HOCINE

CV19T, A NOVEL BIO-SOCIALLY INSPIRED METHOD, BELONGING TO A NEW NATURE-INSPIRED METAHEURISTICS CLASS

Abstract *The paper presents CV19T, a novel bio-socially inspired meta-heuristic, where the cornerstone on which rests is the relationship between humans crowding density, on one side, influenced by their mobility, mutual attractiveness to each other and individual consciousness, and on the other side, the amazing speed of COVID-19 propagation. CV19T originality resides in the fact of combining features from two completely distinct and famous classes, namely: swarm intelligence and Evolutionary Algorithms. Moreover, CV19T extends elitism concept (i.e. survival of the most powerful), on which are based courant evolutionist approaches to the survival of the most beneficial one. Also, CV19T shows that additional parameters can increase control of its behaviour, in many cases, leading to rise in its results relevance. To validate CV19T, it was tested on benchmarks set, including 23 functions (unimodal, multimodal and fixed-dimensional multimodal) and 4 real-world problems.*

Keywords exploration and exploitation, elitism, metaheuristics

Citation Computer Science 25(3) 2024: 351–396

Copyright © 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

Nowadays optimization becomes one of the most important research areas. It covers a wide range of techniques and its fields of application are extremely varied, going from routing in all kinds of networks [20, 59], classification [10, 15, 51], data clustering [5, 47], feature selection [45], air traffic control [22, 53], till choose of economic investments [36]. Systems optimization belonging to such kinds of issues makes it possible to find an adequate configuration, to obtain a gain of effort, time, money, energy, and/or satisfaction. For many of these problems, which must be solvable in first, an optimal solution cannot be found by exact methods, because of the prohibitive execution time due to the excessive size of real-world problems [25]. Stochastic optimization refers to methods employed to minimize or maximize objective functions, in the presence of randomness. They are particularly valuable in addressing complex problems where uncertainty plays a significant role [24].

Unlike traditional optimization approaches, these methods are generally meta-heuristics adopting iterative exploratory strategies in order to increase research efficiency in the resolution space. They are soft computing techniques used to find approximate solutions for hard optimization problems, knowing that soft computing is a term applied to a field of computer science that is characterized by the use of inexact computational solutions, for which an exact solution cannot be find in polynomial time [8, 50]. Their adaptability and robustness make them a strong tool in addressing problems characterized by uncertainty and variability [57]. Nature-inspired meta-heuristics is a very investigated sub-domain of meta-heuristics which derives inspiration from diverse natural resources. For instance, Particle Swarm Optimization (PSO) [27] mirrors the collective behavior of flocks, Artificial Bee Colony (ABC) [33] draws from the intelligent foraging strategies of bee colonies, Genetic Algorithms (GA) [4] replicate evolutionary processes, and Ant Colony Optimization (ACO) [42] emulates cooperative foraging mechanisms observed in ant colonies. Recent advancements in this field, exemplified by algorithms like the Walrus Optimization Algorithm (WaOA) [23], Sewing Training-Based Optimization (STBO) [19], and the Osprey optimization algorithm (OOA) [14], underscore the continual evolution and refinement of these methodologies.

This study is motivated by the increasing in complexity as well as the very demanding nature of current theoretical or real optimization problems. Example: In fields as strategic as telecommunication, space, nuclear and aeronautics/naval construction, where inaccuracies in calculations or delays in responses, once-tolerable, can now days have severe consequences [6, 11]. Also, intrinsic limitations of existing optimization methods (see Sub-section 2.4), considered satisfactory in the near past, begins to require a radical change. Lastly, the extraordinary technological evolution in the field of computer science, especially in terms of calculation power, which becomes more and more parallel [1], opens new horizons encouraging to invent new optimization strategies where focus of researchers should shift towards method efficiency, like: optimizing response time and solution quality, rather than to worry about

number of parameters or their intricacies adjustments or other secondary/technical problems [13]. In response to the above-mentioned incitements, while keeping in mind the already cited technological advances, this study introduces a novel optimization method, named CV19T, drawing inspiration from the transmission dynamics of COVID-19, influenced by a set of main factors, namely: the grouping density of humans, their mobility, their attractiveness to each other and the consciousness specific to each of them. Note that several existing algorithms like CHIO [2] and COVIDOA [35] draw inspiration from COVID-19. However, no one of these algorithms is sufficiently competitive (see sub-Section E).

CV19T is a new bio-socially inspired meta-heuristic dealing with difficult problems in the optimization field. The main contributions of this novel approach are:

- Its features are spread over two completely different main classes of nature-inspired meta-heuristics. This combination of characteristics of two classes, already counting among the most powerful in the field of nature-inspired meta-heuristics, has opened a new axis of very promising investigation and gave CV19T a wide range of possibilities, explored and not explored yet, in screening for the best solutions.
- Switch of CV19T from the restricted principle of “survival of the fittest” on which natural evolution is based to a new one: “survival of the most beneficial”, where the fittest is not always the most beneficial and its removal is, sometimes, most beneficial than its presence, will open new horizons of investigations. In the context of CV19T, this means that elimination of some best individuals from the previous generation will promote the exploration operation and thereby enhancing the algorithm efficiency.
- An increasing in the parameters number of a given nature-inspired meta-heuristic, by adding adequate new ones, can improve, in many cases, the control of its dynamics and therefore improve the efficiency of this method.

In sum, the main objective of this research is to propose a new nature-inspired meta-heuristic, based on innovative ideas that can open up new avenues for research to resolve hard problems with all their complexity, requirements and constraints, which begin to arise in optimization area. In the future, CV19T have to be improved so that it becomes much more efficient than this current version and thus to be able to meet all the already mentioned expectations.

The rest of this paper is organized as follows: Section 2 conducts a comprehensive literature review by exploring existing works in optimization, in general, and nature-inspired meta-heuristics, in more specific. Also, by locating position of the proposed method within nature-inspired meta-heuristics and highlighting research gaps in this research domain. Section 3 presents conceptual foundation of the proposed method, namely CV19T, including assumptions, model description and related investigations. Section 4 outlines CV19T algorithm, discuss parameters tuning details associated to the proposed algorithm and impact of the chosen configurations on exploration and exploitation mechanisms. Section 5 explains the parameters initialization phase,

introduce the used in comparisons benchmarks and the engineering problems. Afterward, it analyses the obtained results and presents their most important managerial insights. Section 6 resumes the treated problem and motivations of this study, summaries the main obtained results and ends, this conclusion section, by suggesting some future outlooks.

2. Literature review

2.1. Optimization problems

Optimization problems have always been part of our daily life and as time passed some, of the well-studied, of them have become benchmarks, used to test a new proposed methods veracity, like traveling salesman problem (TSP), knapsack problem, MAX-SAT problem and many others [40]. In many cases, optimization problems of today's are part of big projects, which leads to a significant increase in their complexity and requirements making them more and more difficult to solve [25].

Optimization can be defined as the process of determining how to use a given resource most effectively possible while still complying with eventual potential limits, linked to this resource. An optimization problem involves, in this case, seeking the solution that most closely approximates to the optimum of a given objective function describing the use of this resource [12].

2.2. Nature-inspired meta-heuristics

Nature-inspired meta-heuristics and their hybrids have revolutionized algorithm design for solving real-world optimization problems. These methods efficiently explore solution spaces, aiming to strike the optimal balance between exploration and exploitation mechanisms to approach, as closely as possible, exact solutions [44]. These approaches have been successful in various fields, providing efficiency, flexibility, and robustness, and was supposed to meet the complex optimization challenges of the future [49].

Meta-heuristic algorithms can be divided into two big main categories, namely: evolutionary algorithms and swarm intelligence-based [41]. The core operation of evolutionary algorithms involves selecting the strongest individuals for survival and discarding the rest [48]. Swarm intelligence is based, for its part, on cooperative behaviours, taking cues from behaviours of social animals like ants/bees colonies, birds flocks, and fishes schools. At the heart of swarm intelligence concept resides synchronization of movements and collective displacements [9]. Another classification, based on inspiration sources, includes: biology [7], physics [31, 32], chemistry [3, 37], sociology [46], and many other natural resources.

Table 1 illustrates meta-heuristic algorithms categorized according to swarm intelligence and evolutionary algorithm classes. Swarm intelligence algorithms featured in Table 1 include Artificial Bees Colonies (ABC), Ants Colonies Optimization (ACO), COOT optimization, Particle Swarm Optimization (PSO), Osprey Optimization Algorithm (OOA), and Walrus Optimization Algorithm (WaOA).

These algorithms draw inspiration from collective behaviours of various animal species and succeeded to solve diverse optimization problems relating to real-world applications. Specifically, ABC is bio-inspired by the behaviours of honeybees, ACO by the foraging behaviours of ants, COOT by the social behaviours of American coot birds, PSO by the collective flight of birds and collective swimming of fishes, OOA by the natural behaviour of ospreys, and WaOA by the behavioural patterns of walruses in nature.

Table 1

Illustration of meta-heuristic algorithms across various categories and historical areas

Algorithm	Ref	Class	Type	Year	Source of inspiration
Particle Swarm Optimization (PSO)	[34]	Swarm-intelligent	Bio-inspired	1980	Swarms
Ant Colony Optimization (ACO)	[16]	Swarm-intelligent	Bio-inspired	1990	Ant colony
Genetic Algorithm (GA)	[18]	Evolutionary Algorithms	Bio-inspired	1996	Evolution
Artificial Bee colony (ABC)	[29]	Swarm-intelligent	Bio-inspired	2005	Honeybee colony
Imperialist Competitive Algorithm (ICA)	[55]	Evolutionary Algorithms	Socially-inspired	2007	Imperialism
Coronavirus herd immunity optimizer (CHIO)	[2]	Evolutionary Algorithms	Socially-inspired	2021	Herd immunity against coronavirus pandemic (COVID-19)
COOT optimization algorithm (COOT)	[39]	Swarm-intelligent	Bio-inspired	2021	American coot birds
Osprey Optimization Algorithm (OOA)	[14]	Swarm-intelligent	Bio-inspired	2022	The natural behavior of osprey
Coronavirus Optimization Algorithm (COVIDOA)	[35]	Evolutionary Algorithms	Bio-inspired	2022	Coronavirus disease replication lifecycle
Sewing Training-Based Optimization (STBO)	[19]	Evolutionary Algorithms	Socially-inspired	2023	Sewing training
Walrus Optimization Algorithm (WaOA)	[23]	Swarm-intelligent	Bio-inspired	2023	Walrus behaviors in nature

The evolutionary algorithms showcased in Table 1 include the Coronavirus Herd Immunity Optimizer (CHIO), which is socially inspired, and COVIDOA, which is bio-inspired. CHIO focusses on optimizing parameters related to herd immunity and vaccine distribution, drawing inspiration from social behaviours and practices related to public health. In contrast, COVIDOA is inspired by biological processes associated with the spread and evolution of the COVID-19 virus, aiming to optimize parameters related to the development and distribution of COVID-19 vaccines. Finally, STBO takes inspiration from the social context of sewing-training. GA and ICA, two other evolutionary algorithms featured in Table 1, are based on principles of evolution and competition.

In general, Table 1 presents a selection of meta-heuristic algorithms from different historical eras and sources of inspiration, providing a glimpse of the diversity and complexity of these optimization methods.

2.3. Position of the proposed method

In this study, the authors introduce a novel bio-socio-inspired meta-heuristic called CV19T. It belongs simultaneously to the class of evolutionary algorithms nature-inspired meta-heuristics, as is the case of the famous GA (Genetic Algorithm) method and to the class of swarm-intelligence-based nature-inspired meta-heuristics, as is the case of the famous PSO (Particle Swarm Optimization) method. This makes it exceptionally original, since it benefits from the features of these two very important classes.

1. **Evolutionary Algorithms class:** This first class is characterized by a set of initial populations, constituting the first generation, which will evolve, along a series of generations, to converge, via the considered method (GA for example), towards the best possible solution. In the case of CV19T, each generation contains only one population. By drawing a parallel between CV19T and methods of this first class, it is clear that the population of CV19T is composed of a set of individuals at each generation, who scan the search space progressively. After each well-determined period, individuals among the infected are eliminated and other uninfected are added. This constitutes a new generation which will try to evolve rapidly, towards the best possible solution. The operations used in this case are:

- **SMB (Survival to the Most Beneficial).** This operation means that every individual within the population can be chosen for removal if it is infected and not beneficial to the community, even if he is among the best/fittest.

So, in contrast to common practice, certain valuable solutions discovered by infected individuals are deliberately selected for deletion rather than preservation. SMB eliminates these good or promising solutions to give more freedom and slack to the exploration mechanism to express itself without having a negative effect on the exploitation mechanism since several of the best-found solutions are still present.

- **RP (Refreshment and Preservation).** This operation combines refreshment of the population and preservation of its size, which reinforces both exploration and exploitation because the regeneration of the new individuals could be either in promise area or sown randomly within the boundaries of the research space.
- **Contagion.** This operation converts the states of the targeted individuals to pass from the susceptible or normal states to the infected one. With regard to all the infected individuals, this represents a recruitment of newly infected individuals who will come to reinforce the exploitation by intensifying the search around the promising solutions.

2. **Swarm-intelligence-based class:** This second class is characterized by a coordination of individuals' movements, belonging to a given swarm, with a view to scrutinizing the search space in a collective and intelligent manner. This makes it possible to bring out a phenomenon of cooperation at the macrolevel produced because of interactions and exchanges of information between the individual's collective at the micro-level. This search policy increases greatly the convergence chances of the considered method towards the best possible solution and in the shortest possible time.

PSO remains one of the most interesting examples in this perspective where each particle moves taking into consideration the position of the particle located on the global optimum found so far and the local optimum of its neighbourhood. Similarity with CV19T is direct; individuals of the considered swarm manage to coordinate their movements with respect to each other by bringing out a phenomenon of collective displacement at macroscopic level. This displacement takes place in two phases: any healthy individual is attracted by an infected individual if it is within its scope. On the other hand, this normal individual can resist this attraction by using his own consciousness force. In the second phase, infected individuals are all attracted to the most infected individual and can also resist its attraction force by using their own consciousness force; which amply reinforces the exploitation.

At the macroscopic level, the observer understands that all these individuals are working in a cooperative manner to converge on the most optimal solution. An individual that is not within the scope of any infected individual is free to move randomly according to the Levy random walk, adapted to the human model, which enhances exploration. The PSO acts in a similar way; each particle must take into account, in its displacement, the position of the best global solution and the best local solutions found in its vicinity.

Also, CV19T draws inspiration from the spread rate of COVID-19 and its correlation with human behaviours. The method incorporates various concepts, such as employing Levy flight to model human displacement, utilizing a stochastic state transformation model (SIR – susceptible, infected, removed), in addition to the two opposing

forces, attractiveness and consciousness, which are mentioned above. Table 2 summarizes the main characteristics and functionalities of CV19T.

Table 2
Recapitulation of main characteristics and functionalities of CV19T

Characteristics		
Name	Description	Inspiration source
Multi-inspiration sources	Leverage strengths of two famous approaches	Swarm Intelligence and Living beings' evolution and adaptation
COVID-19 transmission	Very rapid spread of COVID-19 among humans' gatherings; the more the grouping density of humans increases, the more the propagation speed increases too	Contagious disease transmission and sociology
Balanced research	Good balancing between exploration and exploitation during the search	Nature-inspired meta-heuristics
Attractiveness	Each individual within the population could be attracted to other individuals	Psycho-sociology
Awareness	Force that leads individuals to avoid others in an effort to prevent infection	Psycho-sociology
Humans random walk	Following Levy flight for human displacement	Sociology
Functionalities/Operations		
Name	Description	Inspiration source
SMB (Survival to the Most Beneficial)	Removing some of the promising solutions (most infected individuals) will enhance exploration, which is supposed to reduce the occurrence frequency of the premature convergence phenomenon	Natural evolution
RP (Refreshment and Preservation)	This operation refreshes the current population while preserving its size. This reinforces both exploration and exploitation because the regeneration of the new individuals could be done in promise areas or elsewhere, where they are spread randomly within the boundaries of the research space. This operation maintains population size and balances between exploration/exploitation	Natural evolution
Contagion	This operation concerns the propagation of an infected state which will convert states of near individuals to an infected one to pass from the susceptible or normal states to the infected one. With regard to all the infected individuals, this represents a kind of recruitment of newly infected individuals who will come to reinforce the exploitation by intensifying the search around promising solutions	Contagious disease transmission
Collective displacement	Movements coordination of individuals constituting the considered swarm, with a view to scrutinizing the search space in an efficient manner	School fishes, birds flock, . . . , collective intelligence

2.4. Research gaps

This section focuses on the main shortcomings of nature-inspired meta-heuristics used in the study presented in this research work, knowing that these inadequacies concern most nature-inspired meta-heuristics proposed thus far:

1. Despite their parameterization, the resolution scope of nature-inspired meta-heuristics is limited. In many cases, modeling the solution of a given problem using elements provided by a given meta-heuristic can be challenging or even impossible, like in the case of genetic algorithms when it comes to represent the genes in relation with the problem solution.
2. Solutions discovered by nature-inspired meta-heuristics tend to be overly approximate in some cases.
3. Response times can be significant in many instances.
4. Proposed methods are often limited and can only handle problems instances whose sizes are less to a given threshold. For instance, in the case of the Traveling Salesman Problem (TSP), only a certain number of cities can be addressed before the software ceases to function.
5. Quality of the solution found is proportional to the response time consumed to discover it. Developing an approach to decouple solution quality from response time would be an important scientific breakthrough.
6. It is obvious that a large number of parameters associated with a given method increases its complexity. But this can help, also, in many cases, to control its dynamics, leading thereby to increase its effectiveness. Tempting to control these dynamics in all possible cases will be a good target to reach.
7. Current meta-heuristics lack mechanisms leading to control exploration and exploitation effectively with a view to achieving a more efficiency.
8. Relying on a fixed strategy and constant parameter values restricts the scope of considered nature-inspired meta-heuristics in terms of the problems addressed. Having a dynamic search strategy with adaptable parameter values for different phases of a given meta-heuristic would be highly beneficial.
9. There is a significant research gap in the development of meta-heuristics that integrate inspiration from diverse life areas, such as biology, sociology, psychology, and economy.
10. There is a real lack of theoretical study in the field of meta-heuristics in general and those inspired by nature in particular.
11. Current approaches predominantly focus on one class of meta-heuristics, such as those based on swarm intelligence or evolutionary population dynamics, missing opportunities to leverage synergies arising from combining these different classes.
12. While COVID-19 currently serves as a notable source of inspiration in the field of nature-inspired meta-heuristics, its concepts remain somewhat vague, presenting numerous gaps that hinder more effective exploitation by researchers. For example, statistics studies suggest that various factors, such as blood type, play

a predominant role in COVID-19 spread among humans. However, this remains to be developed further. Once conducted, these studies will likely reinforce the power of meta-heuristics inspired by COVID-19 and may even lead to the proposal of more potent methods.

3. Model description and assumptions

3.1. Effect of people grouping density on infection spread

In the distant past, ancestors of humans lived in small isolated groups, minimizing the spread of infectious diseases unless other agents acted as transmitters. This era, known as “Age of Disaster”, encompassed over 99.99% of humanity’s history. However, around a hundred thousand years ago, the discovery of agriculture and animal husbandry led to increased human grouping marking the onset of the age of diseases [28]. Despite significant strides in controlling infectious diseases, the 20th century witnessed ongoing threats such as SARS, influenza, antimicrobial-resistant bacteria, Ebola, and measles resurgence [26].

More recently, the emergence of new coronavirus diseases as COVID-19, has underscored the great threat of infectious diseases [17, 54]. It is important to note, for this purpose, that COVID-19 primarily spreads through respiratory droplets and aerosols, emphasizing the importance of maintaining a safe distance from others to mitigate transmission [38].

To show the effect of gatherings on the spread of COVID-19, a stochastic transmission process was studied by expanding the Susceptible-Infected-Removed (SIR) model to human behaviours [30].

In this perspective, the studied population is separated into three compartments: susceptible, infected, and removed, and the stochastic state transformation model SIR is applied.

In Figure B1, the flow allowed to pass from one state of transformation to another is shown. Each hour, transitions from the susceptible state to the infected are computed, with a certain probability while taking into consideration two parameters: the infected rate and the area congestion.

The overall area consists of three zone types: crowded zone, mid zone, and uncrowded zone, each with a different probability of infection defined by the number of people gathered there. The risk of infection at each hour is determined according to Equation (2).

$$P_{infection}(t, d) = P_{zone}(t) R_{infected}(d) \quad (1)$$

$$R_{infected}(d) = \frac{N_{infected}(d)}{N_{infected}(d) + N_{Susceptible}(d)} \quad (2)$$

$P_{zone}(t)$ represents the infection probability at time t in the considered zone, 2% is chosen for $P_{crowdedzone}$, 0.2 % for $P_{mid-zone}$, and 0.02 % for $P_{uncrowdedzone}$.

$R_{infected}(d)$ denotes the percentage of infected by the total number of infected and susceptible by day (d).

“Infected” transfers to “removed” with a certain probability every day, knowing that $R_{removed}(d) = 10\%$ [30].

In the development of CV19T, these probabilities are kept the same as in the original research work on COVID-19 [30]. This decision will ensure that CV19T will conform to the dynamics relating to the concrete infection which happen in the real-world.

3.2. Levy flights for human mobility

Levy flight is the ideal method to look for a target in an unfamiliar area randomly. Many experiments have found that the typical characteristics of Levy flights have been shown in the flight (displacement) behaviour of many animal species and insects. Levy flights are, in general, a sort of random walk, where the steps follow Levy’s law of distribution, which can be expressed formally, as shown in Equation (3).

$$L(s) \sim |s|^{-\beta} \quad (3)$$

where s is the step size and $1 < \beta \leq 3$ is an index.

Various studies have shown that human mobility can be defined in terms of Levy flight patterns. A distribution of consistent displacements with Levy flights for human mobility, with $\beta = 1.75 \pm 0.15$, was discovered by Gonzalez and colleagues [21]. This consists of analyzing the mobile phone traces of a set of 10^5 individuals and recording their position every time they receive a phone call or a message.

Levy’s walk pattern is based on two main actions, choice of direction, and step length. In CV19T, authors used the Mantegna Levy flight [56], as formulated below:

$$steplenght = \frac{u_j^{1/\beta}}{v_j} \quad (4)$$

$$v_j = \sigma \cdot randn[D] \quad (5)$$

$$u_j = randn[D] \quad (6)$$

$Randn[D]$ represents the normal distribution of dimension D . The parametre σ represents the variance, and it can be calculated as follow:

$$\sigma = \frac{\Gamma(\beta + 1) \cdot \sin(\pi \cdot \beta/2)}{\Gamma(\frac{1+\beta}{2}) \cdot \beta \cdot 2^{\frac{\beta-1}{2}}} \quad (7)$$

The parameter Γ represents the Gamma function.

According to the previous equation, the new position calculated basing on levy walk pattern is as follow:

$$x_i^{t+1} = x_i^t + \lambda \cdot steplenght \quad (8)$$

Where λ is the scale factor.

3.3. Exploration and exploitation

Broadly speaking, meta-heuristic algorithms have two basic mechanisms of search to know: exploration and exploitation [44]. Exploration involves searching for solutions in areas of the search space not yet explored, whereas exploitation involves intensify research for solutions in the vicinity of promising regions [52]. In the proposed algorithm, there are 4 possible cases: global exploration, local exploration, global exploitation and local exploitation. These four strategies have been visualized in Figure B2, employing a 2D search space for simplicity, where X and Y axes depict the coordinates of both already existing individuals and those which have just been generated within the search space. In what follows, we will explain each of these 4 cases separately:

1. **Case 1 (Global Exploration).** If individual i is uninfected and not associated with any zone, then it moves randomly (see Figure B2 Case 1) by using levy walk through Equation (8).
2. **Case 2 (Local Exploration).** If individual i is uninfected and belongs to an infected zone, then with probability 0.5, an individual j is selected randomly from this same zone to be used in an updating operation of individual i through Equation (9). As evident from the illustration in Figure B2 case 2, the newly generated solutions lie outside the zone in question. This behaviour mimics the tendency of an individual aiming to evade infection by moving away from the infected area. Moreover, Equation (9) highlights that the primary point of attraction is the origin.

$$\text{individual}_i^{n+1}.\text{Position} = (\text{attractor}.\text{Attractiveness} - \text{attracted}.\text{Awareness}) \cdot (\text{attractor}.\text{Position} - \text{attracted}.\text{Position}) \quad (9)$$

The attractor is defined as the individual with the best fitness value between individual i and individual j , while the remaining individual is referred to as the attracted.

3. **Case 3 (Global Exploitation).** If individual i is infected, then it is updated by taking into consideration the Gbest. In this case, attractor is the Gbest, while individual i plays the role of the attracted, and it is updated through Equation (10). The newly generated individuals emerge within the search space between individual i and Gbest as seen in Figure B2 case 3.

$$\text{individual}_i^{n+1}.\text{Position} = \text{individual}_i^n.\text{Position} + \text{Step} \quad (10)$$

the parameter $STEP$ is the amount of displacement amount of individual_i toward its attractor and it is calculated by using Equation (11).

$$\text{Step} = (\text{attractor}.\text{Attractiveness} - \text{attracted}.\text{Awareness}) \cdot (\text{attractor}.\text{Position} - \text{attracted}.\text{Position}) \quad (11)$$

The parameter *Awareness* is the prevention measures taken by those which are attracted. The parameter *Attractiveness* is the attraction force of the attractor.

$$Attractivness = (\alpha \cdot (|attractor.Cost + tran|/|attracted.Cost + tran|)) \quad (12)$$

The parameter α is used to balance between *Attractiveness* and awareness. The parameter *tran* is a transition, and it used to prevent too small or too big values for *Attractiveness* and it can be calculated as:

$$tran = \max(|attractor.Cost, attracted.Cost|) + 1 \quad (13)$$

4. **Case 4 (Local Exploitation).** If individual i is uninfected within a given infected zone, there is a probability of 0.5 that this individual will follow the local best (Lbest) attractor, following Equation (10). In this context, the individual is considered as the attracted party. It is important to note that the emergence of newly generated individuals occurs within the research space between individual i and its attractor, as depicted in Figure B2 under “case 4”.

3.4. Spatial grouping

As seen in Figure B3, each infected individual is a center of a circle, called the infected zone.

Individuals within each infected zone are chosen by calculating the Euclidean distance between the above-mentioned infected individual and the rest of the population, knowing that only those who are located inside the aforesaid circle of *IR* ray are taken into account.

By using the *IR* (Infection Rayon), update the zone that each individual belongs to. Note that lower *IR* enhances the diversity, while higher *IR* enhances convergence since more individuals will enter the infected zone of an infected individual which will increase the number of individuals moving toward their infector (see Sub-section 4.2).

The *IR* can be calculated by using Equation (14):

$$IR = MaxDistance \cdot P \quad (14)$$

While *distance* is a decimal value between 0 and 1, high *distance* means large *IR*, while low *distance* means small *IR*. The maximum distance is calculated by using Equation (15):

$$MaxDistance = Distance(Max, min) \quad (15)$$

Max and *Min* are tow vectors of size d , designating respectively *UpperBound* and *LowerBound*.

$$Max = [UB, UB, ...]^d, Min = [LB, LB, ...]^d \quad (16)$$

If an individual is present in two or more zones, it is considered to belong to the zone where the distance to the infector is the shortest among all the zones in question.

The type of a given zone is contingent upon the number of individuals within this zone. For example, a zone containing less or equal to 2 individuals is denoted as an “uncrowded zone”, while a zone containing between 3 and 10 individuals is classified as a “mid-zone”. Conversely, a zone containing more than 10 individuals is designated as a “crowded zone”. These numerical thresholds are adaptable to specific requirements. For example, by elevating the minimum threshold for the categorization of a zone like the one “crowded” the likelihood of potential infection transmission decreases.

This is entirely justifiable, given that the probability of infection transmission is notably higher within crowded zones compared to moderately or slightly overcrowded zones.

In this research, the aforementioned numerical thresholds were meticulously selected through an iterative process involving various permutations. The chosen numerical configuration aligns appropriately with the population size used in this study, containing fifty individuals.

3.5. Assumptions

Assumptions on what the proposed method principle is based can be described as follows:

- COVID-19 virus spreads very rapidly among humans.
- High density of human groupings increases the speed of COVID-19 propagation. There is a proportional relationship between them; higher clustering density leads to increased virus spread.
- Individuals can be attracted to each other for various psycho-sociological reasons: curiosity, inherent human nature of sociability, personal interest towards attractors (e.g., a fan), etc. In this work, what matters is that susceptible individuals are attracted by infected individuals.
- The more individuals are aware of the danger, the more they tend to avoid it; thus, the force of consciousness opposes the force of attraction.
- Human movement, resembling a random pattern to an external observer, adheres to the well-defined mathematical principles of Levy flights.
- Each infected individual has an infection zone with a given range. The solutions space is thus structured into so-called infection zones with different clustering densities.
- Individuals who do not belong to any infection zone are considered as part of the non-infected zone.
- An individual can belong to one or more infection zones or none. If he belongs to multiple infection zones, he is assigned to the one with the closest attractor to him. If the distance is the same for all, one of these infected zones will be selected randomly.

- There are different cases of individual movement including:
 - A susceptible individual located outside all the infection zones will move randomly according to Levy's random walk law adapted to the human model.
 - A susceptible individual located inside a given infection zone will try to flee this zone based on their level of awareness to avoid the infection.
 - An infected individual moves according to the influence exerted on him by the most infected individual of all infection zones (the Gbest).

4. Solution approach

In this section, authors elucidate the strategies employed to address optimization challenges using the proposed nature-inspired meta-heuristic. This segment outlines the algorithm's procedural steps, offering a comprehensive guide for its implementation. Additionally, the discussion includes parameters tuning, ensuring adaptability and robustness across diverse problems.

4.1. Steps of CV19T

1. **Initialization phase:** apart from the parameters fine-tuned in the dedicated Subsection (see Sub-section 4.2), crucial general parameters like population size (NP) and iteration number should be initialized at the beginning.
2. **Evaluation:** In this phase, the fitness function is employed to assess the considered population in order to identify the initial infected individuals. The set of these infected individuals, whose cardinality is IN, contains the highest fitness values, and the highest value among them is saved as Gbest.
3. **End of day:** If (IterationNumber mod DaySize) is different from 0 go to step 7: *this verifies if the day ends or not.*
4. **Remove randomly 10% of infected individuals:** this quantity represents the probability of the removed individuals and is noted $R_{removed}(d)$. To avoid the extinction of the considered population, the removed number of individuals is regenerated in such a way that some of them surround the GBest and the others are spread randomly over the whole of the search space. The state of the regenerated individuals is "susceptible".
5. **Update infection probabilities, via Equation (1):** the infection probability of a given individual depends on the zone type in which it belongs and the total number of infected individuals.
6. **Change the state of the AIN best individuals in the population to "infected":** to show the propagation of the disease, this algorithm adds, each day, a certain number of infected individuals from the best ones.
7. **Zones creation:** The process of zone creation is accomplished by considering each infected individual as the centre of a circle, constituting what we call "infected zone". Individuals within each infected zone are selected based on their Euclidean distance from the infected individual; only those inside the considered

circle, of Rayon “IR”, are taken into account. The number of individuals present in a given zone determines its type. This flexible classification, based on specific needs, aids to understand infection transmission probabilities within each zone type (refer to Sub-Section 3.4 for details).

8. **Update infection state:** In the context of a given population, an individual's state can be either “infected” or “susceptible”. The transition from a susceptible state to an infected state is determined through the application of probabilities outlined in step 5. This is done by updating the infection status of each individual within its population. It should be noted that individuals who do not belong to any zone, are classified as part of “uncrowded zone”.
9. **Update position:** CV19T is inspired by human displacement causing a crowding phenomenon, facilitating the spread of COVID-19. This displacement is characterized by four distinct cases, each representing a unique strategy for adjusting an individual's position, as detailed in Sub-Section 3.3.

The exploration process focuses on susceptible individuals and occurs in two ways: firstly, for those who are not assigned to any specific zone and move randomly using the Levy walk law, and secondly, for individuals within an infection zone, who randomly select another individual from the same zone and adapt their search strategy accordingly.

The exploration goal is to enhance the global search by diversifying the search in order to attempt to reach a comprehensive coverage of the search space.

On the other hand, exploitation involves some infected-individuals, attracted by the global best and who agree to be attracted (i.e. their own consciousness forces goes in the same direction of the attraction force exercised on them), and also some susceptible-individuals within each infection zone, which are attracted by their infector and who agree to undergo this attractive force

10. **Evaluation:** it consists of using the adequate fitness function, which depends on the problem to solve, in order to evaluate the population, after the updating position of each of its individuals, and return the new global best (GBest), if it exists, and changes its state to “infected” if its state is “susceptible”.
11. **Stop:** This last phase involves the verification conditions forcing the algorithm to halt. Once these conditions are satisfied, this algorithm provides the best solutions found, before stopping. Conversely, if the specified conditions are not met yet, the algorithm will revert to step 3 and resume its iterative execution. The expression of the termination conditions is: “a predefined maximum number of iterations and one of the following sub-conditions: a convergence towards a fixed solution or achievement of an acceptable optimal solution”.

The whole process is summarized in Algorithm C3 (see Appendix).

4.2. CV19T parameters tuning

This sub-section concerns parameters of the suggested method (CV19T). Some of these parameters are invented by the authors, which know nothing about their val-

ues. So, this sub-section explains the conducted strategy to find the adequate values of these parameters and show the adopter values of the others parameters already given by other researchers. CV19T’s parameter tuning embraces diverse strategies. Key parameters like Levy flight exponent faithfully mirror their real-world counterparts, such as the human random walk pattern, for an authentic representation of natural phenomena. For core elements like the infection probability, CV19T leverages insights gleaned from already established research on COVID-19 transmission rates [30].

This ensures the behaviour of infected individuals, within the algorithm, accurately reflects the virus’s spread in the real world, lending a valuable layer of realism and potentially harnessing the intricate dynamics observed in viral propagation.

Additionally, certain parameters are intentionally fixed, in the beginning, by the authors, offering flexibility for adjustments based on the population size. Notably, parameters like the initial infected number ($IN = 1$), day size in term of hours ($DaySize = 24$) and Added Infected Number ($AIN = 3$) are set, initially, by the authors and can be fine-tuned depending on the population size.

Another subset of parameters undergoes a training process, including *Distance*, α *LowerBound* (LB), α *UpperBound* (UB), *Awareness LB*, *Awareness UB*, λ *LB* and λ *UB*, emphasizing the adaptability and precision of CV19T in the capture of the complex-optimization-landscapes dynamics.

For the adjustment of CV19T algorithm parameters, various settings are considered, presented as scenarios (refer to Table 3).

Table 3
Parameters tuning scenarios

		Distance	α LB	α UB	Awareness LB	Awareness UB	λ LB	λ UB
Scenario 1	Awareness	0.1	0	2	0	1	1	2
Scenario 2		0.1	0	2	1	2	1	2
Scenario 3		0.1	0	2	0	0.1	1	2
Scenario 4		0.1	0	2	0	0.01	1	2
Scenario 5		0.1	0	2	0	0.001	1	2
Scenario 6	λ	0.1	0	2	0	0.1	0	2
Scenario 7		0.1	0	2	0	0.1	0	1
Scenario 8		0.1	0	2	0	0.1	1	2
Scenario 9	α	0.1	0	2	0	0.1	1	2
Scenario 10		0.1	0	1	0	0.1	1	2
Scenario 11		0.1	1	2	0	0.1	1	2
Scenario 12	P	0.01	1	2	0	0.1	1	2
Scenario 13		0.3	1	2	0	0.1	1	2
Scenario 14		0.7	1	2	0	0.1	1	2
Scenario 15		1	1	2	0	0.1	1	2

To select the best parameters values, the CV19T algorithm is tested on 6 benchmarks (see Sub-Section 5.2). Among these benchmarks, 2 are unimodal (F1, F6), 2 are multimodal (F10, F12), and 2 are fixed-dimensional (F14, F19). Evaluation of each scenario involves the execution of the concerned algorithm 5 times in a row,

and this independently, to mitigate the impact of chance on the results. Subsequently, the mean value is calculated to facilitate comparison. This methodology empowers the authors to ascertain the most suitable parameter configuration for CV19T algorithm.

Table 4 presents the outcomes obtained by applying different parameters scenarios to the targeted benchmarks. The results indicate that scenario 3 yields the most favourable results in terms of *Awareness*. Consequently, the appropriate range for the *Awareness* parameter (*Awareness*) falls between 0 and 0.1.

Regarding the λ parameter, which serves as the step size for Levy walk displacement, scenario 8 exhibits the optimal parameter configuration, with λ ranging between 1 and 2.

For the α parameter, scenario 11 demonstrates the best performance, suggesting that α should be set within the range of 1 to 2.

Table 4
Parameters tuning results

		F01	F06	F10	F12	F14	F19
Scenario 1	Awareness	1.60E+03	2.35E+03	1.52E+01	2.01E+06	3.55E+00	-3.85E+00
Scenario 2		2.14E+04	2.54E+04	1.47E+01	1.39E+08	8.77E+00	-3.54E+00
Scenario 3		3.52E+02	3.82E+02	1.28E+01	7.24E+01	5.33E+00	-3.86E+00
Scenario 4		1.41E+02	6.56E+02	1.65E+01	5.55E+03	8.21E+00	-3.86E+00
Scenario 5		2.69E+02	5.76E+02	1.48E+01	4.89E+01	7.80E+00	-3.86E+00
Scenario 6	α	7.92E+02	2.76E+03	1.19E+01	6.40E+02	7.27E+00	-3.86E+00
Scenario 7		8.61E+02	1.35E+03	1.46E+01	2.20E+03	9.69E+00	-3.86E+00
Scenario 8		2.19E+02	6.51E+02	1.46E+01	6.87E+01	8.34E+00	-3.86E+00
Scenario 9	λ	1.97E+02	3.80E+02	1.54E+01	2.23E+04	6.10E+00	-3.86E+00
Scenario 10		1.07E+04	1.48E+04	1.59E+01	2.32E+07	4.36E+00	-3.83E+00
Scenario 11		1.12E+02	2.34E+02	1.90E+01	1.24E+02	5.31E+00	-3.86E+00
Scenario 12	P	1.56E+02	6.35E+01	1.88E+01	2.87E+03	6.30E+00	-3.86E+00
Scenario 13		3.14E+02	3.22E+02	1.76E+01	6.81E+02	1.17E+01	-3.86E+00
Scenario 14		0.00E+00	1.68E-03	8.88E-16	4.56E-04	3.17E+00	-3.86E+00
Scenario 15		0.00E+00	1.69E-03	8.88E-16	1.67E-04	1.39E+00	-3.86E+00

In the case of the *Distance* parameter used to define the infection radius, the value of 1 is considered appropriate. Figure 1 showcases the exploration and exploitation behaviour of CV19T algorithm in different scenarios. In this representation, the X-axis denotes the iteration count, whereas the Y-axis denotes the count of individuals engaged in exploration and exploitation during each iteration.

Scenarios 1 to 13 show an imbalance between the exploration and exploitation mechanisms. Conversely, scenarios 14 and 15 exhibit an interesting balance between these two mechanisms, which helps to elucidate the results presented in Table 4.

Furthermore, the parameter *P* exerts a significant influence over the equilibrium between exploration and exploitation. This observation is drawn from the analysis of Scenarios 12, 13, 14, and 15.

Specifically, it is discernible that when P assumes smaller values (as depicted in Figure 1 for Scenarios 12 and 13), a greater proportion of individuals within the system engage in exploration activities, as evidenced by the corresponding curves. However, as the value of parameter P is increased, it is noted a growth in the number of individuals participating in both exploration and exploitation. This rise is, notably, accompanied by a prevalence of oriented exploitation activities (see the Scenarios 14 and 15 in Figure 1).

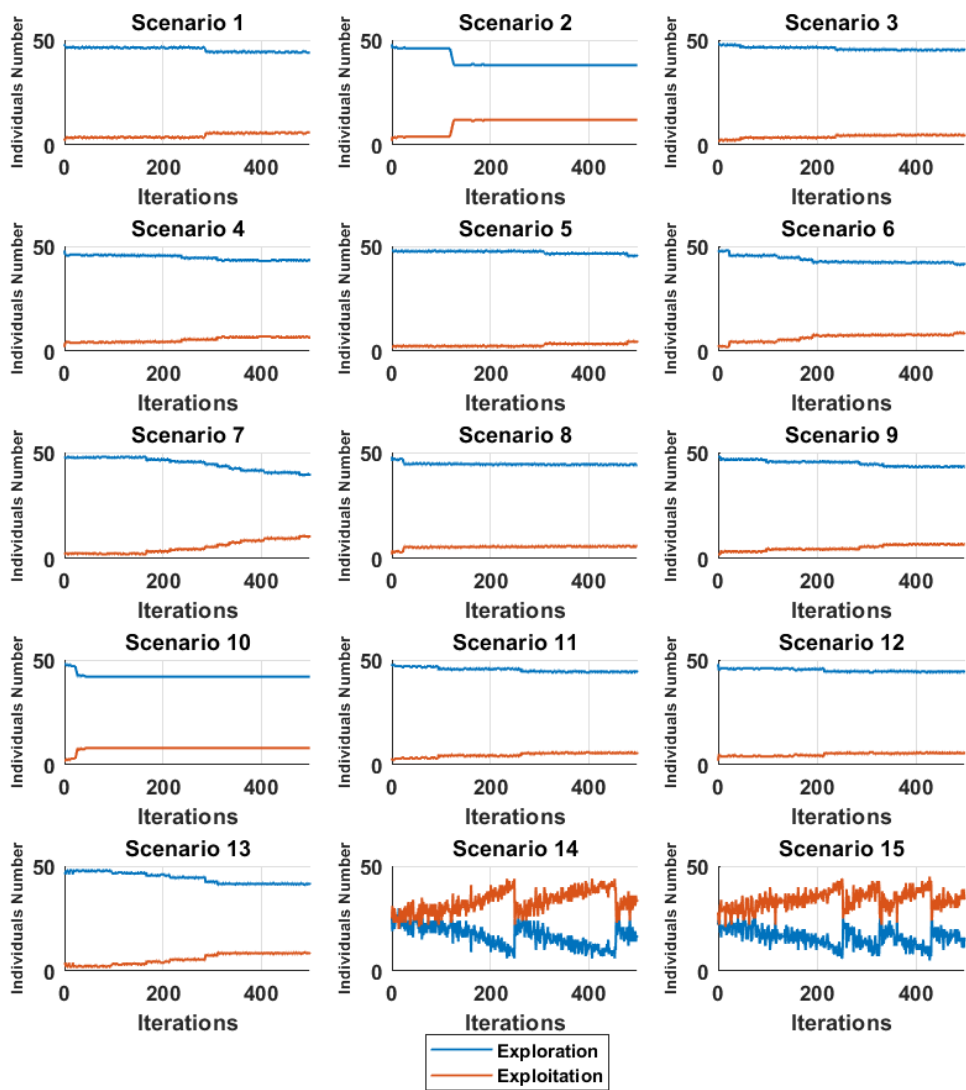


Figure 1. Exploration and exploitation rates basing on different settings

5. Results and discussion

Data analysis plays a pivotal role in assessing the performance and effectiveness of meta-heuristic algorithms, particularly when comparing and evaluating their outcomes. By employing rigorous data analysis techniques, we gain insights into the strengths and weaknesses of each meta-heuristic, facilitating a comprehensive understanding of their behavior across diverse scenarios. Furthermore, the application of data analysis is essential in discerning the meta-heuristic's adaptability to both standardized benchmark functions and real-life engineering problems, providing valuable information for researchers and practitioners to make informed decisions on algorithm selection and implementation.

This section encompasses the results derived from applying the proposed method to both benchmark functions and real-life engineering problems. The comprehensive data analysis not only unveils the algorithm's performance metrics but also provides managerial insights, offering practical implications for decision-makers in choosing and implementing meta-heuristic algorithms in various applications.

5.1. Parameter settings for the 8 algorithms used in comparison

To assess the competitiveness of the proposed meta-heuristic, a comparative analysis was conducted against 8 meta-heuristics originating from various historical areas and different sources of inspiration. This sub-section concerns parameters of the 8 chosen reference-methods for the results comparisons stage; values of these parameters are already established by other researchers. These reference-methods included ABC (Artificial Bee Colony), ACO (Ant Colony Optimization), CHIO (Coronavirus Herd Immunity Optimizer), COOT, COVIDOA (COVID-19 Inspired Optimization Algorithm), GA (Genetic Algorithm), ICA (Imperialist Competitive Algorithm), and PSO (Particle Swarm Optimization).

For each of these meta-heuristics, two types of parameters were considered: general parameters and algorithm-specific parameters. General parameters encompass factors such as population size, number of iterations, and problem dimension. In these conditions, the general parameter values were set as follows: population size = 50, iteration number = 10.000, and problem dimension (applicable to a non-fixed dimensional problems) = 30.

To ensure a fair comparison, each algorithm was executed 20 times in a row, and this independently of each other. Specific settings, for each algorithm, were outlined in Table A1 (see Appendix). Note that all specific parameters referring to CV19T algorithm were gathered in Sub-section 4.2.

5.2. Benchmark functions

Unimodal benchmarks are mathematical functions with only one global optimum, this means that only one point in the function's domain gives the highest value. Such functions are helpful in testing the exploitation ability of nature-inspired meta-heuristics,

i.e. their ability to converge towards the global optimum. On the other hand, multimodal benchmarks are mathematical functions with multiple local optima, this means that various points in the function's domain can give relatively high values. These functions are helpful in testing the exploration ability of optimization algorithms, i.e. their ability to find diverse solutions and avoid getting trapped in a given local optima.

By using both unimodal and multimodal benchmarks, researchers can assess the overall performance of targeted optimization algorithms by evaluating their ability to converge to the global optimum while avoiding getting trapped in any local optimum.

In this work, CV19T is tested using 7 unimodal benchmarks (see Appendix Figure D1), 6 multimodal (see Appendix Figure D2) and 10 fixed-dimensional multimodal benchmarks (see Appendix Figure D3). Fixed-dimensional multimodal benchmarks encompass multi-modal benchmark functions characterized by unchanging dimensions. These benchmarks are also harnessed to evaluate the efficacy of exploration strategies.

5.2.1. Results for benchmark functions

Comparative outcomes of the proposed algorithm in relation with 8 other methods are showcased in Tables E1, E2, E3 and E4 (see Appendix). The results are expressed in statistical terms of the best (BEST), mean (MEAN), worst (WORST), standard deviation (STD) values and have been analysed by using the statistical approach named: "two-tailed t-test". During the t-test analysis, results of the proposed method are compared with each of the aforementioned reference algorithms. The outcome of this comparison is assessed by using null hypothesis H0 and alternative hypothesis H1. H0 refers to the null hypothesis and means that no significant difference exists between the compared algorithms. On the other hand, H1 refers to the alternative hypothesis, which goes against the hypothesis H0. In this practical phase, the two hypothesis forms were utilized.

Table E1 presents the outcomes obtained by using unimodal benchmarks to assess the exploitation ability of the proposed algorithm. It is evident from table E1 that CV19T algorithm achieved the highest results across all the unimodal benchmarks, demonstrating its exceptional performance. Notably, CV19T successfully reached the exact solution in 5 out of 7 benchmarks, providing strong evidence of the proposed algorithm's robust exploitation capability. The COOT algorithm is in the second position by obtaining the best results in 4 out of 7 benchmarks. However, CV19T exhibits superior performance in terms of standard deviation (STD) values, which indicates that it is more suitable for this kind of optimisation problems. According to t-test results, it is obvious that CV19T behaviours are very close to those of COOT, while they differ greatly from those of ACO and CHIO.

Table E2 presents the outcomes obtained from testing multimodal benchmarks to assess the exploration ability of the proposed algorithm. Based on the obtained results, CV19T shows its superiority on the other methods in the resolution of the

6 considered benchmarks. ICA followed in second position with a score of 4 out of 6 benchmarks, while sharing with CV19T the two highest performances. Among the 6 used benchmarks, CV19T was found to be the more suitable in three functions (F09, F10, and F11), while ICA was deemed more suitable in the remaining benchmarks. COOT algorithm came in third position, sharing the highest scores in 3 benchmarks with CV19T, but it was considered, overall, the less suitable. The remaining algorithms exhibited similar medium performances, except for COVIDOA and GA, which was the less successful.

Results of the two-tailed t-test indicate that there is a significant difference in performance between CV19T and the other algorithms. The alternative hypothesis H1 is supported for most algorithms across all benchmarks, stipulating that their own performances differ significantly from those of CV19T, except that CV19T exhibits a noticeable similarity with PSO.

For the fixed-dimensional multi-modal benchmarks, the results obtained by the various algorithms appear to be closely aligned. More precisely, CV19T, COOT, ICA, and PSO achieve the best performances for all the benchmarks (10/10). Notably, CV19T and COOT exhibit higher suitability based on the MEAN WORST and STD results. The remaining algorithms performed also well, securing a more than good rankings (with 9 out of 10 benchmarks, except of F15). The COVIDOA algorithm, however, came last by succeeding only in 8 out of 10 benchmarks.

Two-tailed t-test allowed to come out with the similarities, in matter of performances, between CV19T and ACO (7 out of 10 benchmarks). Conversely, CV19T demonstrated significant dissimilarity with both CHIO and COVIDOA, highlighting thereby the distinctiveness of this algorithm from the other algorithms inspired of COVID-19.

These findings shed light on the hard competitiveness of CV19T, COOT, ICA, and PSO algorithms in achieving top results in solving the targeted benchmarks. Furthermore, the t-test analysis underscores the unique characteristics of CV19T when compared to the other COVID-19-inspired algorithms.

The convergence curves of the compared algorithms relating to the targeted uni-modal and multimodal benchmarks are illustrated, respectively, in Figure F1 and F2 (see Appendix). In this representation, the X-axis corresponds to the iteration count, while the Y-axis signifies the lowest cost discovered during each iteration (the same for Figure F3 and F4). A careful graphs analysis reveals distinct patterns: Firstly, CV19T exhibits an exceptional convergence rate, as its curve remains almost stick to the X-axis along all benchmarks except for F8. This indicates that CV19T converges rapidly towards optimal solutions. On the other hand, ABC and CHIO display a moderate convergence speed, as their curves show a gradual decline towards X-axis. ACO, however, exhibits the slowest convergence speed among all the used algorithms, as its curve takes a longer time to approach the X-axis. The remaining algorithms demonstrate a convergence speed close to that of CV19T, which testify their ability to quickly move towards optimal solutions.

Figure F3 (see Appendix) showcases the convergence curves of the algorithms for fixed dimensional multimodal benchmarks. A comprehensive analysis of the graph reveals several key observations. In most cases, the algorithms exhibit similar convergence speeds. However, for benchmark F21, COOT and ICA display the fastest convergence rates, indicating their ability to quickly approach optimal solutions. Conversely, ACO, GA, and COVIDOA demonstrate the slowest convergence speeds for this benchmark. The remaining algorithms, including CV19T, exhibit a medium level of performance.

Moving on to benchmark F22, ABC, COOT, GA, and ICA stand out with the fastest convergence rates. Conversely, PSO displays the lowest convergence rate among all the algorithms. The rest of the algorithms demonstrate a medium level of convergence performance for this benchmark. For benchmark F23, CV19T, ABC, COOT, COVIDOA, and ICA showcase the fastest convergence speeds, indicating their efficiency in converging towards optimal solutions. On the other hand, ACO and GA exhibit the lowest convergence rates. PSO and CHIO display a medium convergence speed for this particular benchmark.

Figure F3 highlights the varying convergence speeds of the algorithms for fixed dimensional multimodal benchmarks. While some algorithms perform well others are bad, according to the targeted benchmark. Broadly speaking, the algorithms majority show medium convergence performance, with a few outliers demonstrating exceptionally fast or slow convergence rates depending on the considered benchmark.

5.3. Real-life engineering problems

CV19T algorithm was utilized to solve 4 engineering problems, namely: the Three Bar Truss Design (TBTD), Stepped Cantilever Beam Design (SCBD), Multiple Disc Clutch Brake Design (MDCBD), and Hydrodynamic Thrust Bearing Design (HTBD) (see Appendix Figure B4) [58] [43]. In the TBTD problem (see Figure B4a), there are 2 parameters to optimize, namely A1 and A2. In SCBD problem, there are 10 parameters to optimize, as depicted B4b. MDCBD problem has 4 parameters to optimize (see Figure B4c), and finally, HTBD problem has 4 parameters to optimize (see Figure B4d). It is noteworthy that all these problems were formulated as optimization problems to minimize.

5.3.1. Results for real-life engineering problems

Table E5 (see Appendix) illustrates the outcomes obtained when applying CV19T and the other aforementioned methods to solve the 4 engineering problems. Overall, all the algorithms produced close results, except for MDCBD where CV19T clearly outperformed them all, demonstrating thus its superiority.

For the HTBD problem, it is interesting to observe that ICA algorithm delivered the fastest result in terms of the BEST-criterion, but it fails in terms of suitability for other aspects of the problem.

To evaluate the performance of the other algorithms relative to CV19T, a two-tailed t-test was carried out. The obtained results indicate that most algorithms exhibited similar performance to CV19T for the SCBD and HTBD problems, which suggests that they are statistically comparable. However, the CHIO algorithm deviated from this trend to verify the alternative hypothesis. On the other hand, when examining the TBTD and MDCBD problems, most of the reference-algorithms, used in this study, verified the alternative hypothesis, indicating a significant difference in performance when compared with CV19T.

In this perspective, Figure F4 (see Appendix) showcases the curves behaviours of all these reference-algorithms at time of their resolution of all the targeted problems.

It is evident, from Figure F4, that the convergence speeds of TBTD, SCBC, and HTBD are comparable. However, when analysing MDCBD, it becomes apparent that both CV19T and ICA exhibit the most rapid convergence, yielding the most favourable outcomes. On the other hand, COVIDOA displays the slowest convergence speed when solving MDCBD. As for the remaining methods, their convergence speeds are relatively similar.

5.4. Managerial insights

- CV19T application to the 4 targeted engineering problems, producing competing solutions to some famous methods, shows its real efficiency. This practical success underlines that CV19T is already ready to be used to solve real-world problems.
- The proposed approach will open, certainly, a new avenue for combining between features of several distinct classes of nature-inspired meta-heuristics to spawn entirely new meta-heuristic type, leveraging the strength of the combined classes. This operation different entirely from hybridization principal.
- Authors of this research work invented a new operator concerning the evolutionary approaches class, which generalizes the “survival of the fittest” principal to the “survival of the most beneficial” principal. This idea will surely be widely used in future research works relating to meta-heuristics based on the population evolution.
- Authors have shown, in this study, that adding new adequate parameters has increased, in many cases, the dynamics control of CV19T to make it more effective (see Appendix Table A2). In this perspective, they believe that this finding can be generalized by trying to find even better parameter values than those already found. This will open up a new research axis, pushing researchers to try to increase the efficiency of existing meta-heuristics by adding adequate parameters to them and adjusting their values as appropriate.

6. Conclusion

Current optimization problems become increasingly difficult to solve while presenting very hard constraints and requirements that even nature-inspired meta-heuristics

which count between the most performant existing methods to solve them begin to find serious difficulties to deal with. To fill this increasingly gap, the solution addressed in this research work is to suggest a novel bio-socio-inspired meta-heuristic aiming to compete with the famous, powerful or alike type methods in the field of nature-inspired metaheuristics.

Given that the field of nature-inspired metaheuristics is currently the most popular in terms of solving difficult optimization problems, because of the satisfactory results they have provided in the near past, it is obvious that this represents a fairly strong motivation encouraging authors to start their research work by taking this metaheuristics-sub-domain as a starting point. CV19T has proven to be as consistent and efficient as the carefully selected nature-inspired meta-heuristics used in the comparative study. In many cases, conducted statistical analysis revealed the superiority of the proposed algorithm in terms of convergence speed and quality of found solutions, and even it manages to discover, sometimes, the exact solutions. On the other hand, the results related to the experienced engineering problems testify the promising future of CV19T in solving real-world problems. Through this research, the authors have opened a new avenue of investigation by demonstrating the possibility of combining characteristics of two completely different classes of nature-inspired metaheuristics to propose a new powerful method. In addition, the concept of “survival of the most beneficial” generalized to this of “survival of the fittest” will undoubtedly be adopted in future researches relating to the evolutionary approaches. Also, adding new adequate parameters will be a new investigation direction to improve the control and increase the performances of existing meta-heuristics.

In the near future, the authors planned to use CV19T to solve challenging theoretical problems, such as the Traveling Salesman Problem, while trying to surpass the results obtained so far, and also real-world problems, related to domains as strategic as digital image processing, cryptography, wireless networks, collective robotics, and swarms of drones. Refinements and improvements to the CV19T meta-heuristic and more judicious adjustments to its parameters will be necessary in this case.

References

- [1] Ajani S.N., Khobragade P., Dhone M., Ganguly B., Shelke N., Parati N.: Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing, *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12(7S), pp. 546–559, 2024. <https://ijisae.org/index.php/IJISAE/article/view/4159>.
- [2] Al-Betar M.A., Alyasseri Z.A.A., Awadallah M.A., Abu Doush I.: Coronavirus herd immunity optimizer (CHIO), *Neural Computing and Applications*, vol. 33, pp. 5011–5042, 2021. doi: 10.1007/s00521-020-05296-6.
- [3] Alatas B.: A novel chemistry based metaheuristic optimization method for mining of classification rules, *Expert Systems with Applications*, vol. 39(12), pp. 11080–11088, 2012. doi: 10.1016/j.eswa.2012.03.066.

- [4] Alhijawi B., Awajan A.: Genetic algorithms: theory, genetic operators, solutions, and applications, *Evolutionary Intelligence*, vol. 17, pp. 1245–1256, 2024. doi: 10.1007/s12065-023-00822-6.
- [5] Ariyaratne A., Ilankoon I., Samarasinghe U., Silva R.: Finding Playing Styles of Badminton Players Using Firefly Algorithm Based Clustering Algorithms: Finding Playing Styles of Badminton Players Using FA Variants, *Computer Science*, vol. 24(3), 2023. doi: 10.7494/csci.2023.24.3.5116.
- [6] Bhandari S., Sahay K.B., Singh R.K.: Optimization Techniques in Modern Times and Their Applications. In: *2018 International Electrical Engineering Congress (iEECON)*, pp. 1–4, 2018. doi: 10.1109/IEECON.2018.8712308.
- [7] Binitha S., Sathya S.S.: A Survey of Bio inspired Optimization Algorithms, *International Journal of Soft Computing and Engineering*, vol. 2(2), pp. 137–151, 2012. <https://www.ijscce.org/portfolio-item/B0523032212/>.
- [8] Blum C., Roli A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Computing Surveys (CSUR)*, vol. 35(3), pp. 268–308, 2003. doi: 10.1145/937503.937505.
- [9] Chakraborty A., Kar A.K.: Swarm Intelligence: A Review of Algorithms. In: *Nature-Inspired Computing and Optimization: Theory and Applications*, pp. 475–494, Springer, 2017. doi: 10.1007/978-3-319-50920-4_19.
- [10] Chola Raja K., Kannimuthu S.: Deep learning-based feature selection and prediction system for autism spectrum disorder using a hybrid meta-heuristics approach, *Journal of Intelligent & Fuzzy Systems*, vol. 45(1), pp. 797–807. doi: 10.3233/jifs-223694.
- [11] Conway B.A., Paris S.W.: Spacecraft Trajectory Optimization Using Direct Transcription and Nonlinear Programming. In: B.A. Conway (ed.), *Spacecraft Trajectory Optimization*, pp. 37–78, Cambridge Aerospace Series, Cambridge University Press, 2010. doi: 10.1017/CBO9780511778025.004.
- [12] De León-Aldaco S.E., Calleja H., Alquicira J.A.: Metaheuristic optimization methods applied to power converters: A review, *IEEE Transactions on Power Electronics*, vol. 30(12), pp. 6791–6803, 2015. doi: 10.1109/tpel.2015.2397311.
- [13] Deb K.: Multi-objective optimisation using evolutionary algorithms: an introduction. In: L. Wang, A.H.C. Ng, K. Deb (eds.), *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, pp. 3–34, Springer, London, 2011. doi: 10.1007/978-0-85729-652-8_1.
- [14] Dehghani M., Trojovský P.: Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems, *Frontiers in Mechanical Engineering*, vol. 8, 1126450, 2023. doi: 10.3389/fmech.2022.1126450.
- [15] Dhief I., Feroskhan M., Alam S., Lilith N., Delahaye D.: Meta-Heuristics Approach for Arrival Sequencing and Delay Absorption Through Automated Vectoring. In: *2023 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2023. doi: 10.1109/cec53210.2023.10254077.

- [16] Dorigo M., Birattari M., Stützle T.: Ant colony optimization, *IEEE Computational Intelligence Magazine*, vol. 1(4), pp. 28–39, 2006. doi: 10.1109/MCI.2006.329691.
- [17] Fan Y., Zhao K., Shi Z.L., Zhou P.: Bat coronaviruses in China, *Viruses*, vol. 11(3), 210, 2019. doi: 10.3390/v11030210.
- [18] Forrest S.: Genetic algorithms, *ACM Computing Surveys (CSUR)*, vol. 28(1), pp. 77–80, 1996. doi: 10.1145/234313.234350.
- [19] Ghadimi N., Yasoubi E., Akbari E., Sabzalian M.H., Alkhazaleh H.A., Ghadam-yari M.: SqueezeNet for the forecasting of the energy demand using a combined version of the sewing training-based optimization algorithm, *Heliyon*, 2023. doi: 10.1016/j.heliyon.2023.e16827.
- [20] Gomes R., Vieira D., de Castro M.F.: Application of Meta-Heuristics in 5G Network Slicing: A Systematic Review of the Literature, *Sensors*, vol. 22(18), 6724, 2022. doi: 10.3390/s22186724.
- [21] González M.C., Hidalgo C.A., Barabási A.L.: Understanding individual human mobility patterns, *Nature*, vol. 458(7196), pp. 779–782, 2008. doi: 10.1038/nature07850.
- [22] Grishin I.Y., Timirgaleeva R.R.: Air navigation: Optimisation control of means cueing of the air-traffic control system. In: *2017 21st Conference of Open Innovations Association (FRUCT)*, pp. 134–140, IEEE, 2017. doi: 10.23919/fruct.2017.8250175.
- [23] Han M., Du Z., Yuen K.F., Zhu H., Li Y., Yuan Q.: Walrus optimizer: A novel nature-inspired metaheuristic algorithm, *Expert Systems with Applications*, vol. 239, 122413, 2023. doi: 10.1016/j.eswa.2023.122413.
- [24] Hannah L.A.: Stochastic optimization. In: J.D. Wright (ed.), *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*, pp. 473–481, Elsevier London, England, 2015. doi: 10.1016/b978-0-08-097086-8.42010-6.
- [25] Harsha P., Charikar M., Andrews M., Arora S., Khot S., Moshkovitz D., Zhang L., et al.: Limits of Approximation Algorithms: PCPs and Unique Games (DIMACS Tutorial Lecture Notes), *arXiv preprint arXiv:10023864*, 2010. doi: 10.48550/arXiv.1002.3864.
- [26] Hoang T., Coletti P., Melegaro A., Wallinga J., Grijalva C.G., Edmunds J.W., Beutels P., Hens N.: A systematic review of social contact surveys to inform transmission models of close-contact infections, *Epidemiology (Cambridge, Mass)*, vol. 30(5), 723, 2019. doi: 10.1097/ede.0000000000001047.
- [27] Jain M., Saihjpal V., Singh N., Singh S.B.: An overview of variants and advancements of PSO algorithm, *Applied Sciences*, vol. 12(17), 8392, 2022. doi: 10.3390/app12178392.
- [28] Jones S.: *Germs, Genes and Genesis: The History of Infectious Disease*, Gresham College, 2016.
- [29] Karaboga D.: Artificial bee colony algorithm, *Scholarpedia*, vol. 5(3), 6915, 2010. doi: 10.4249/scholarpedia.6915.

- [30] Karako K., Song P., Chen Y., Tang W.: Analysis of COVID-19 infection spread in Japan based on stochastic transition model, *Bioscience trends*, 2020. doi: 10.5582/bst.2020.01482.
- [31] Kaveh A., Akbari H., Hosseini S.M.: Plasma generation optimization: a new physically-based metaheuristic algorithm for solving constrained optimization problems, *Engineering Computations*, 2020. doi: 10.1108/ec-05-2020-0235.
- [32] Kaveh A., Bakhshpoori T.: Water evaporation optimization: a novel physically inspired optimization algorithm, *Computers & Structures*, vol. 167, pp. 69–85, 2016. doi: 10.1016/j.compstruc.2016.01.008.
- [33] Kaya E., Gorkemli B., Akay B., Karaboga D.: A review on the studies employing artificial bee colony algorithm to solve combinatorial optimization problems, *Engineering Applications of Artificial Intelligence*, vol. 115, 105311, 2022. doi: 10.1016/j.engappai.2022.105311.
- [34] Kennedy J., Eberhart R.: Particle swarm optimization. In: *Proceedings of ICNN'95 – International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, 1995. doi: 10.1109/ICNN.1995.488968.
- [35] Khalid A.M., Hosny K.M., Mirjalili S.: COVIDOA: a novel evolutionary optimization algorithm based on coronavirus disease replication lifecycle, *Neural Computing and Applications*, vol. 34(24), pp. 22465–22492, 2022. doi: 10.1007/s00521-022-07639-x.
- [36] Krasovskii A.A., Taras'ev A.M.: Dynamic optimization of investments in the economic growth models, *Automation and Remote Control*, vol. 68(10), pp. 1765–1777, 2007. doi: 10.1134/S0005117907100050.
- [37] Lam A.Y.S., Li V.O.K.: Chemical-reaction-inspired metaheuristic for optimization, *IEEE Transactions on Evolutionary Computation*, vol. 14(3), pp. 381–399, 2009. doi: 10.1109/tevc.2009.2033580.
- [38] Lotfi M., Hamblin M.R., Rezaei N.: COVID-19: Transmission, prevention, and potential therapeutic opportunities, *Clinica Chimica Acta*, vol. 508, pp. 254–266, 2020. doi: 10.1016/j.cca.2020.05.044.
- [39] Naruei I., Keynia F.: A new optimization method based on COOT bird natural life model, *Expert Systems with Applications*, vol. 183, 115352, 2021. doi: 10.1016/j.eswa.2021.115352.
- [40] Parejo J.A., Ruiz-Cortés A., Lozano S., Fernandez P.: Metaheuristic optimization frameworks: a survey and benchmarking, *Soft Computing*, vol. 16, pp. 527–561, 2012. doi: 10.1007/s00500-011-0754-8.
- [41] Piotrowski A.P., Napiorkowski M.J., Napiorkowski J.J., Rowinski P.M.: Swarm intelligence and evolutionary algorithms: Performance versus speed, *Information Sciences*, vol. 384, pp. 34–85, 2017. doi: 10.1016/j.ins.2016.12.028.
- [42] Rezvanian A., Mehdi Vahidipour S., Sadollah A.: An Overview of Ant Colony Optimization Algorithms for Dynamic Optimization Problems. In: M. Andriychuk, A. Sadollah (eds.), *Optimization Algorithms – Classics and Recent Advances*, IntechOpen, Rijeka, 2023. doi: 10.5772/intechopen.111839.

- [43] Şahin İ., Dörterler M., Gökçe H.: Optimization of Hydrostatic Thrust Bearing Using Enhanced Grey Wolf Optimizer, *Mechanika*, vol. 25(6), pp. 480–486, 2019. doi: 10.5755/j01.mech.25.6.22512.
- [44] Saib B., Abdessemed M.R., Hocin R., Khoualdi K.: Study of Exploration and Exploitation Mechanisms in Nature Inspired Metaheuristics for Global Optimization. In: M.R. Laouar, V.E. Balas, B. Lejdel, S. Eom, M.A. Boudia (eds.), *12th International Conference on Information Systems and Advanced Technologies "ICISAT 2022"*, pp. 442–453, Springer International Publishing, Cham, 2023. doi: 10.1007/978-3-031-25344-7_41.
- [45] Sayed S.A.F., ElKorany A., Sayed S.: Applying hunger game search (HGS) for selecting significant blood indicators for early prediction of ICU Covid-19 severity, *Computer Science*, vol. 24(1), pp. 113–136, 2023. doi: 10.7494/csci.2023.24.1.4654.
- [46] Shastri A., Nargundkar A., Kulkarni A.J.: A Brief Review of Socio-inspired Metaheuristics. In: *Socio-Inspired Optimization Methods for Advanced Manufacturing Processes*, pp. 19–29, Springer Series in Advanced Manufacturing, Springer, Singapore, 2021. doi: 10.1007/978-981-15-7797-0_2.
- [47] Shial G., Sahoo S., Panigrahi S.: A Nature Inspired Hybrid Partitional Clustering Method Based on Grey Wolf Optimization and JAYA Algorithm, *Computer Science*, vol. 24(3), pp. 361–405, 2023. doi: 10.7494/csci.2023.24.3.4962.
- [48] Slowik A., Kwasnicka H.: Evolutionary algorithms and their applications to engineering problems, *Neural Computing and Applications*, vol. 32, pp. 12363–12379, 2020. doi: 10.1007/s00521-020-04832-8.
- [49] Stork J., Eiben A.E., Bartz-Beielstein T.: A new taxonomy of global optimization algorithms, *Natural Computing*, vol. 21, pp. 219–242, 2022. doi: 10.1007/s11047-020-09820-4.
- [50] Talbi E.G.: *Metaheuristics: from design to implementation*, John Wiley & Sons, 2009. doi: 10.1002/9780470496916.
- [51] Tantithamthavorn C., McIntosh S., Hassan A.E., Matsumoto K.: Automated parameter optimization of classification techniques for defect prediction models. In: *ICSE'16: Proceedings of the 38th International Conference on Software Engineering*, pp. 321–332, 2016. doi: 10.1145/2884781.2884857.
- [52] Tilahun S.L.: Balancing the Degree of Exploration and Exploitation of Swarm Intelligence Using Parallel Computing, *International Journal on Artificial Intelligence Tools*, vol. 28(03), 1950014, 2019. doi: 10.1142/s0218213019500143.
- [53] Visintini A.L., Glover W., Lygeros J., Maciejowski J.: Monte Carlo optimization for conflict resolution in air traffic control, *IEEE Transactions on Intelligent Transportation Systems*, vol. 7(4), pp. 470–482, 2006. doi: 10.1109/tits.2006.883108.
- [54] Wang C., Horby P.W., Hayden F.G., Gao G.F.: A novel coronavirus outbreak of global health concern, *The Lancet*, vol. 395(10223), pp. 470–473, 2020. doi: 10.1016/s0140-6736(20)30185-9.

- [55] Xing B., Gao W.J.: Imperialist Competitive Algorithm. In: *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms. Intelligent Systems Reference Library*, pp. 203–209, Springer, Cham, 2014. doi: 10.1007/978-3-319-03404-1_15.
- [56] Yang X.S.: *Nature-inspired metaheuristic algorithms*, Luniver Press, 2010.
- [57] Yang X.S., He X.: Swarm Intelligence and Evolutionary Computation: Overview and Analysis. In: X.S. Yang (ed.), *Recent Advances in Swarm Intelligence and Evolutionary Computation*, Studies in Computational Intelligence, vol. 585, pp. 1–23, Springer, Cham, 2015. doi: 10.1007/978-3-319-13826-8_1.
- [58] Yin S., Luo Q., Du Y., Zhou Y.: DTSMA: Dominant swarm with adaptive t-distribution mutation-based slime mould algorithm, *Mathematical Biosciences and Engineering*, vol. 19(3), pp. 2240–2285, 2022. doi: 10.3934/mbe.2022105.
- [59] Zhang H., Wang X., Memarmoshrefi P., Hogrefe D.: A survey of ant colony optimization based routing protocols for mobile ad hoc networks, *IEEE Access*, vol. 5, pp. 24139–24161, 2017. doi: 10.1109/access.2017.2762472.

Appendix

A. Tables

Table A1
Algorithms setting

Algorithm	Settings
ABC	Number of Onlooker Bees=population size Trial Limit=round($0.6 \times \text{number of desecion variable} \times \text{population size}$) Acceleration Coefficient Upper Bound=1
ACO	Sample Size=40 Selection Pressure=0.5 Deviation-Distance Ratio=1
CHIO	Number of solutions have corona virus=1 Spreading rate parameter=0.05
COOT	Number of leaders=ceil($0.1 \times \text{population size}$) Number of coot=population size- Number of leaders
COVIDOA	shifting Number=1 number Of Subprotiens=2 MR=0.1 Gamma=0.5 Beta=0.5
GA	Crossover Percentage=0.7 Number of Offsprings (also Parnets)= $2 \times \text{round}(\text{Crossover Percentage} \times \text{population size} / 2)$ Extra Range Factor for Crossover=0.4 Mutation Percentage=0.3 Number of Mutants=round($\text{Mutation Percentage} \times \text{population size}$) Mutation Rate=0.1
ICA	Number of Empires/Imperialists=10 Selection Pressure=1 Assimilation Coefficient=1.5 Revolution Probability=0.05 Revolution Rate=0.1 Colonies Mean Cost Coefficient=0.2
PSO	Inertia Weight=1 Inertia Weight Damping Ratio=0.99 Personal Learning Coefficient=1.5 Global Learning Coefficient=2 Inertia Weight=1

Table A2 presents the comparative results of CV19T metaheuristic with and without *Attractiveness* and *Consciousness* parameters for specific benchmarks used in the considered research work; these benchmarks are those who support the hypothesis that adding new adequate parameters can increase the dynamic control of the considered meta-heuristic, leading to increase the quality of its solutions.

Numerical values, shown it the Table A2, represent the performance metrics for aforementioned specific benchmarks (F06, F10, F12, F15, F23). For the configuration without *Attractiveness* and *Consciousness*, the results indicate higher values, signifying a larger objective function value in the optimization process (while remembering that it's a matter of minimization problems). In contrast, when *Attractiveness* and *Consciousness* parameters are incorporated, the values notably decrease, indicating improved performance with enhanced control over the metaheuristic's behavior. Authors of this work believe strongly that for the other benchmarks they can obtain the same results but they have to readjust the parameters values, under this new hypothesis, to find more adequate ones.

Table A2
Parameter sensitivity in CV19T: A closer look at
Attractiveness and *Consciousness* effects on specified functions

	F05	F06	F10	F12	F15	F23
Without <i>Attractiveness</i> and <i>Consciousness</i>	9.99E+02	2.49E+02	8.88E−16	1.32E−02	4.16e−04	−9.99E−00
With <i>Attractiveness</i> and <i>Consciousness</i>	2,14E+00	1.69E−03	8.88E−16	1.67E−04	3.08E−04	−1.1E+01

B. Figures

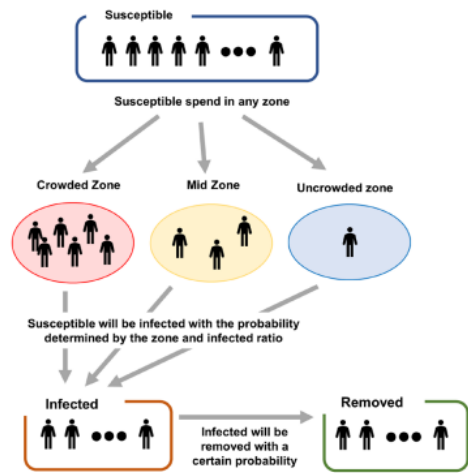


Figure B1. Transition flow between the three compartments:
Susceptible, Infected and removed

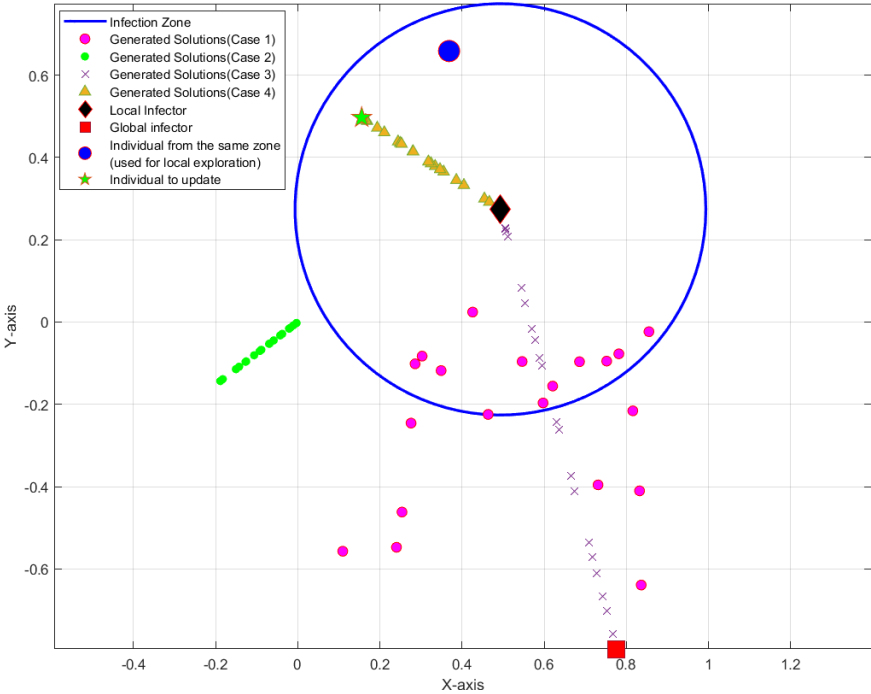


Figure B2. Visualization of the Four Search Strategies in the Proposed Algorithm within a 2D Search Space

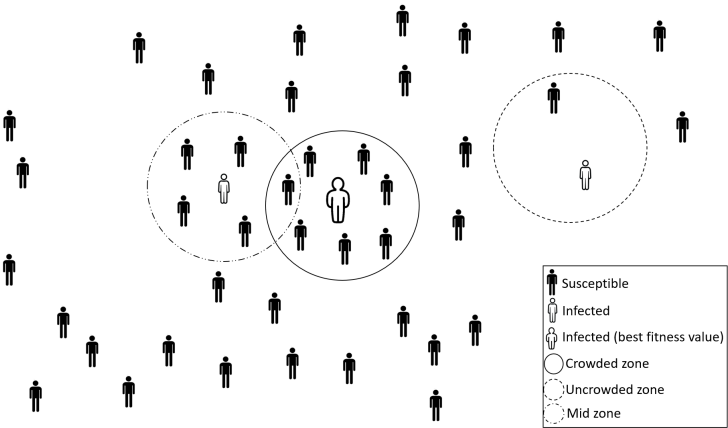


Figure B3. Zone creation

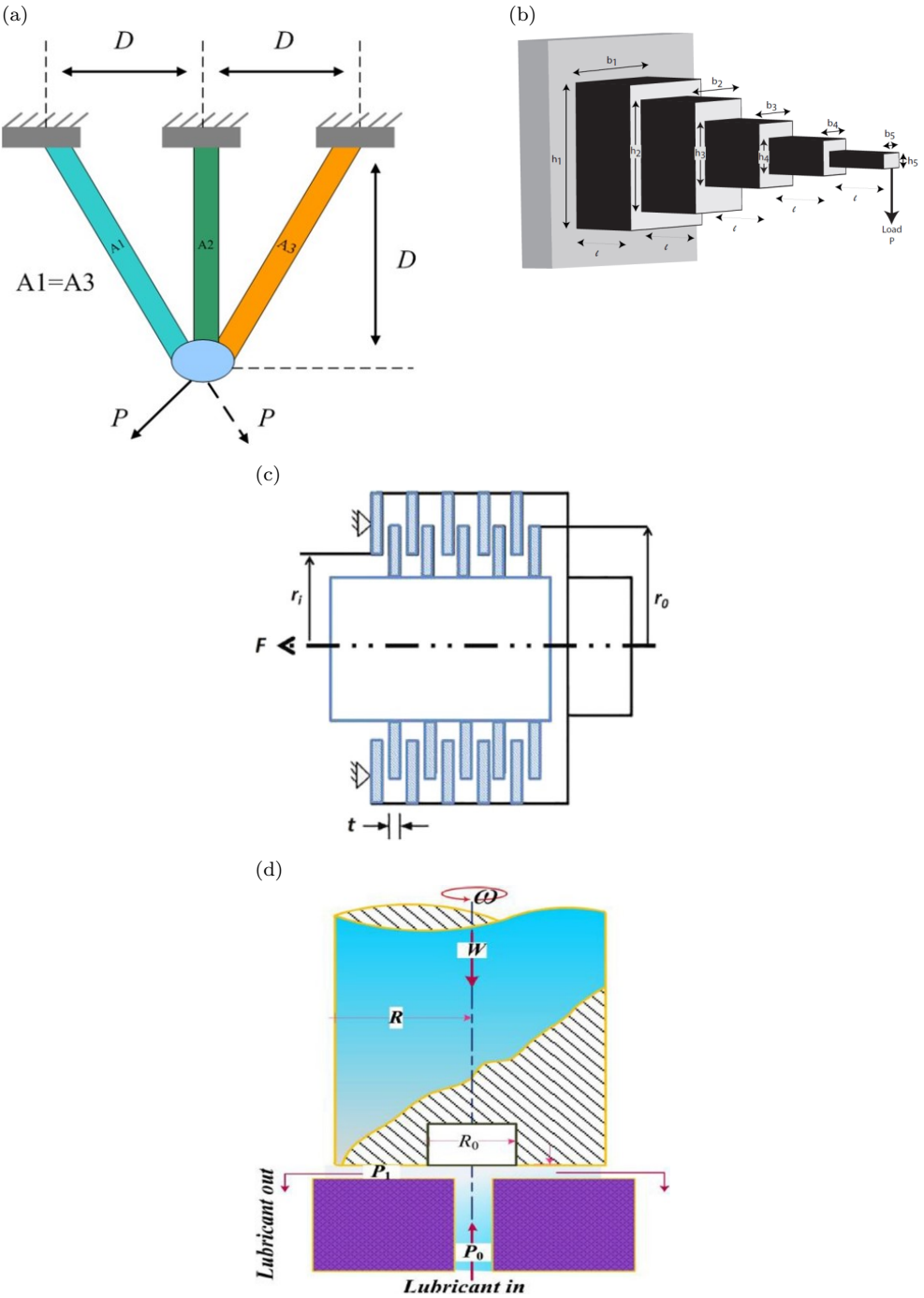


Figure B4. Real-life engineering problems

C. Algorithms

Algorithm C1: Zone creation

```

1 Initialize distance parameter;
2 calculate IR using Equation (15);
3 for  $i=1:1:IN$  do
4   for  $j=1:1:populationSize$  do
5      $dist = EuclideanDistance(individual_i, Individual_j)$ 
6     if  $individual_j \neq individual_i$  and  $dist \leq IR$  and  $individual_j.Zone = 0$  then
7        $individual_j.Zone = individual_i$ 
8        $individual_j.dist = dist$ 
9     else
10      if  $individual_j \neq individual_i$  and  $dist \leq IR$  and  $dist \leq individual_j.dist$  and
11         $individual_j.Zone \neq 0$  then
12         $individual_j.Zone = individual_i$ 
13         $individual_j.dist = dist$ 
14      end if
15    end if
16  end for
17 end for

17 for  $i=1:1:IN$  do
18    $BN = \text{the number of individual belonging to } individual_i.IR$ 
19   if  $BN \geq 10$  then
20      $individual_i.ZoneType = CrowdedZone$ 
21   else
22     if  $BN \geq 2$  and  $BN < 10$  then
23        $individual_i.ZoneType = MidZone$ 
24     else
25        $individual_i.ZoneType = UncrowdedZone$ 
26     end if
27   end if
28 end for

```

Algorithm C2: Update Position

```

1 if  $individual_i.Zone = 0$  and  $individual_i.Stat = 0$  then
2   Case 1;
3 else
4   if  $individual_i.Stat = 1$  then
5     Case 3;
6   else
7     if  $rand() < 0.5$  then
8       Case 2;
9     else
10      Case 4;
11    end if
12  end if
13 end if

```

Algorithm C3: CV19T

```

1 Initialization of parameters;
2 Randomly generate initial population ;
3 Evaluate initial population;
4 Return gbest;
5 Set the IN number of individuals as infected;
  // IN is the number of infected individuals

6 while stop criteria doesn't achieved do
7   if (the day ends) then
8     for i=1:1:IN do
9       if (rand <  $R_{removed}$ ) then
10         /*  $R_{removed}=10$  is the probability of removing infected individuals */
11         Remove individual i;
12         if (rand < 0.5 then
13           | Re-generate individual i randomly;
14         else
15           | Re-generate individual i surround gBest;
16         end if
17       end if
18       Update infection probabilities;
19     end for
20     Change the state of the best AIN of the population to "infected";
21     // AIN is the added number of infected individuals
22   else
23     Zones creation;
24     Update infection state;
25     Update position;
26     Evaluation;
27     if (IN >  $PpulationSize/2$ ) then
28       | Change the state of all individuals to not-infected;
29     end if
30   end if
31 end while

32 Return best solution;

```

D. Benchmark functions

The *Range* parameter represents the Lower and the Upper Bounds of the research area, n represents the dimensions of the function, C represents the nature of the Benchmark (U for unimodal and M for Multimodal) and $F(x^*)$ represents the global optimum.

Table D1
Unimodal benchmark functions

Function	Equation	Range	n	C	$F(x)^*$
F1	$\sum_{i=1}^n x_i^2$	$[-100, 100]$	30	U	0
F2	$\sum_{i=1}^n x_i + \prod_{i=1}^n x_i $	$[-10, 10]$	30	U	0
F3	$\sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2$	$[-100, 100]$	30	U	0
F4	$\max_{1 \leq i \leq n} x_i $	$[-100, 100]$	30	U	0
F5	$\sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$[-30, 30]$	30	U	0
F6	$\sum_{i=1}^{n-1} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]$	30	U	0
F7	$\sum_{i=1}^n ix_i^4 + \text{rand}[0, 1]$	$[-128, 128]$	30	U	0

Table D2
Multimodal benchmark functions

Function	Equation	Range	n	C	$F(x)^*$
F8	$\sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	$[-500, 500]$	30	M	$-12, 569.5$
F9	$\sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	30	M	0
F10	$-20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}) - \exp(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)) + 20 + \exp(1)$	$[-32, 32]$	30	M	0
F11	$\frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	30	M	0
F13	$0.1 \left(\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 (1 + \sin^2(3\pi x_{i+1})) + (x_n - 1)^2 (1 + \sin^2(2\pi x_n)) \right) + \sum_{i=1}^n u(x_i, 5, 100, 4)$	$[-50, 50]$	30	M	0

Table D3
Fixed-dimensional multimodal benchmark functions

Function	Equation	Range	n	C	$F(x)^*$
F14	$\left(0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right)^{-1}$	$[-65, 65]$	2	M	1
F15	$\sum_{i=1}^{11} \left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right)^2$	$[-5, 5]$	4	M	0.00030
F16	$4x_1^2 - 2.1x_1^4 + \frac{x_1^6}{3} + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]$	2	M	-1.0316
F17	$\left(x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos(x_1) + 10$	$[-5, 5]$	2	M	0.398
F18	$\left(1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right) \times \left(30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \right)$	—	—	—	—
F19	$-\sum_{i=1}^4 c_i \exp\left(-\sum_{j=1}^3 a_{ij}(x_j - p_{ij})^2\right)$	$[-1, 3]$	3	M	$-3, 86$
F20	$-\sum_{i=1}^4 c_i \exp\left(-\sum_{j=1}^6 a_{ij}(x_j - p_{ij})^2\right)$	$[0, 1]$	6	M	-3.32
F21	$-\sum_{i=1}^5 \left((x - a_i) \cdot (x - a_i)^T + c_i \right)^{-1}$	$[0, 10]$	4	M	-10.1532
F22	$-\sum_{i=1}^7 \left((x - a_i) \cdot (x - a_i)^T + c_i \right)^{-1}$	$[0, 10]$	4	M	-10.4028
F23	$-\sum_{i=1}^{10} \left((x - a_i) \cdot (x - a_i)^T + c_i \right)^{-1}$	$[0, 10]$	4	M	-10.5363

E. Results tables

Table E1
Uni-modal benchmarks results

	CV19T	ABC	ACO	CHIO	COOT	COVIDOA	GA	ICA	PSO
F01	BEST	0,0000E+00	5.263E-24	1.000E-05	1.608E-40	0,0000E+00	8.104E+00	6.768E-01	1.156E-121
	MEAN	0,0000E+00	1.457E-22	2.676E-05	1.268E+00	1.587E-130	2.023E+01	3.264E+00	1.357E-21
	WORST	0,0000E+00	8.848E-22	7.133E-05	4.014E+00	3.174E-129	3.898E+01	7.344E+00	2.712E-20
	STD	0,0000E+00	2.104E-22	1.602E-05	1.325E+00	7.097E-130	1.718E+00	9.420E-47	6.064E-21
F02	H	✓	1	1	1	0	0	0	1
	BEST	0,0000E+00	1.210E-29	7.022E-16	2.313E-01	0,0000E+00	4.490E-01	2.570E-31	1.908E-05
	MEAN	0,0000E+00	1.940E-28	7.508E-12	3.235E-01	1.309E-258	6.338E-01	1.039E-28	4.430E-03
	WORST	0,0000E+00	7.490E-28	1.327E-10	3.869E-01	2.617E-257	1.141E+00	6.797E-01	2.472E-02
F03	STD	0,0000E+00	1.970E-28	2.969E-11	4.361E-02	0,0000E+00	1.638E-01	2.693E-28	6.412E-03
	H	✓	0	0	0	0	0	0	0
	BEST	0,0000E+00	1.559E+04	4.087E+04	1.866E+03	0,0000E+00	9.687E+01	1.152E+02	4.338E-38
	MEAN	0,0000E+00	2.025E+04	6.530E+04	2.815E+03	4.9e-324	3.582E+02	1.900E+02	6.760E-34
F04	WORST	0,0000E+00	2.453E+04	9.035E+04	4.152E+03	4.9e-323	5.629E+01	1.565E+03	8.044E-33
	STD	0,0000E+00	2.170E+03	1.234E+04	6.765E+02	0,0000E+00	2.212E+02	3.460E+03	2.000E-33
	H	✓	1	1	0	1	1	1	1
	BEST	0,0000E+00	4.313E+01	3.506E+00	2.069E+00	0,0000E+00	1.079E+00	1.389E+00	3.030E-19
F05	MEAN	0,0000E+00	4.703E+01	5.747E+00	3.483E+00	1.824E-179	1.569E+00	1.922E+00	5.371E-17
	WORST	0,0000E+00	5.034E+01	1.012E+01	4.983E+00	3.647E-178	2.311E+00	2.372E+00	2.729E-16
	STD	0,0000E+00	2.478E+00	1.838E+00	8.947E-01	0,0000E+00	3.436E-01	2.435E-01	9.494E-17
	H	✓	1	1	0	1	1	1	1
F06	BEST	2,3841E-10	1.940E+01	2.281E+01	2.004E+00	2.121E+01	8.612E+01	1.376E+02	1.546E-05
	MEAN	2,1352E+00	2.057E+01	2.316E+01	8.823E+01	2.274E+01	9.795E+02	3.001E+02	2.803E+01
	WORST	2,1365E+01	2.356E+01	2.347E+01	1.495E+02	2.369E+01	3.296E+03	5.344E+02	7.443E+01
	STD	6.572E+00	7,7601E-01	1,7523E-01	4.715E+01	6,4638E-01	8.746E+02	1.054E+02	2.279E+01
F07	H	✓	1	1	1	0	1	0	1
	BEST	0,0000E+00	0,0000E+00	7.709E-25	0,0000E+00	1.376E-13	1.578E+00	7.459E-01	9.553E-32
	MEAN	0,0000E+00	9.815E-32	6.039E-24	4.122E-01	1.463E-11	4.019E+00	3.102E+00	1.403E-21
	WORST	0,0000E+00	1.196E-30	1.671E-23	8.722E-01	1.295E-10	7.797E+00	6.094E+00	1.550E-20
F08	STD	0,0000E+00	2.730E-31	4.793E-24	3.257E-01	3.081E-11	1.686E+00	1.086E+00	4.254E-21
	H	✓	0	1	1	1	1	1	1
	BEST	4,5388E-07	2.816E-02	1.170E-02	2.233E-02	1.374E-05	2.298E-03	3.378E-03	4.497E-04
	MEAN	6,1099E-06	4.123E-02	2.067E-02	3.617E-02	1.228E-04	7.662E-03	5.742E-03	1.353E-03
F09	WORST	2,1113E-05	5.256E-02	3.442E-02	4.936E-02	1.347E-04	1.460E-02	1.011E-02	2.648E-03
	STD	5,1119E-06	8.003E-03	5.757E-03	8.323E-03	7.984E-05	3.303E-03	2.190E-03	6.151E-04
	H	✓	1	1	1	0	1	1	0

Table E2
Multi-modal benchmarks results

	CV19T	ABC	ACO	CHIO	COOT	COVIDO	GA	ICA	PSO
F08	BEST	-1,257E+04	-5,516E+03	-1,257E+04	-1,045E+04	-1,038E+04	-1,150E+04	-1,257E+04	-7,713E+03
	MEAN	-1,221E+04	-5,998E+03	-1,244E+04	-9,308E+03	-8,473E+03	-1,150E+04	-1,247E+04	-6,519E+03
	WORST	-1,113E+04	-5,618E+03	-1,220E+04	-8,356E+03	-6,403E+03	-1,067E+04	-1,231E+04	-4,712E+03
	STD	4,837E+02	3,375E+02	1,191E+02	5,255E+02	1,151E+03	2,467E+02	9,211E+01	7,927E+02
	H	1	1	1	1	1	1	1	1
F09	BEST	0,000E+00	1,533E+02	0,000E+00	0,000E+00	5,616E+01	2,273E-01	5,684E-14	1,990E+01
	MEAN	0,000E+00	1,809E+02	1,532E-01	1,165E-13	9,021E+01	2,866E+00	9,452E-01	4,472E+01
	WORST	0,000E+00	2,003E+02	6,684E-01	1,990E-12	1,229E+02	5,748E+00	2,985E+00	8,756E+01
	STD	0,000E+00	9,820E+00	2,194E-01	4,474E-13	1,993E+01	1,770E+00	1,140E+00	1,373E+01
	H	1	1	1	1	1	1	1	1
F10	BEST	8,882E-16	4,441E-15	2,931E-14	8,882E-16	4,621E-01	2,458E-02	2,931E-14	6,839E-14
	MEAN	8,882E-16	7,461E-15	3,618E-02	1,306E-13	7,619E-01	5,128E-01	3,499E-14	1,255E+00
	WORST	8,882E-16	7,994E-15	2,436E-01	2,530E-12	1,310E+00	1,504E+00	4,352E-14	2,739E+00
	STD	0,000E+00	1,302E-15	7,513E-02	5,649E-13	2,185E-01	4,907E-01	5,206E-15	8,025E-01
	H	1	1	1	1	1	1	1	0
F11	BEST	0,000E+00	0,000E+00	9,212E-09	0,000E+00	9,895E-01	3,167E-01	0,000E+00	0,000E+00
	MEAN	0,000E+00	2,466E-02	1,173E-01	5,551E-18	1,037E+00	9,047E-01	3,274E-02	1,203E-02
	WORST	0,000E+00	2,755E-01	6,971E-01	1,110E-16	1,102E+00	1,054E+00	1,075E-01	7,306E-02
	STD	0,000E+00	6,743E-02	2,354E-01	2,483E-17	2,949E-02	1,823E-01	3,262E-02	1,713E-02
	H	1	1	1	0	1	1	1	0
F12	BEST	1,571E-32	8,229E-28	1,571E-32	5,569E-16	2,352E-02	3,950E-04	1,571E-32	1,651E-32
	MEAN	1,433E-30	2,411E-24	1,260E-03	1,489E-13	3,009E-01	3,149E-01	1,571E-32	8,293E-02
	WORST	1,966E-29	2,497E-23	2,482E-03	1,645E-12	2,680E+00	1,120E-02	1,571E-32	3,110E-01
	STD	4,597E-30	5,857E-24	9,161E-04	3,724E-13	5,861E-01	2,663E-03	2,808E-48	1,095E-01
	H	1	1	1	0	0	0	1	0
F13	BEST	1,350E-32	4,634E-20	1,350E-32	7,249E-15	1,032E-01	1,949E-02	1,350E-32	1,572E-30
	MEAN	1,130E-31	1,461E-17	3,000E-02	4,507E-12	2,329E-01	1,168E-01	1,350E-32	1,342E-01
	WORST	1,061E-30	1,013E-16	6,177E-02	4,436E-11	4,842E-01	2,671E-01	1,350E-32	8,975E-01
	STD	2,616E-31	2,449E-17	2,262E-02	1,111E-11	1,031E-01	5,912E-02	2,808E-48	2,641E-01
	H	1	1	1	1	1	1	0	1

Table E3
Fixed dimensional Multi-modal benchmarks results F14–F19

	\	ABC	ACO	CHIO	COOT	COVIDOA	GA	ICA	PSO
F14	BEST	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01
	MEAN	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	3.216E+00
	WORST	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	9.980E-01	1.076E+01
	STD	1.057E-09	0.000E+00	6.553E-11	7.204E-17	2.538E-10	4.875E-15	1.441E-16	2.580E+00
	H	1	1	1	1	1	1	1	1
F15	BEST	3.075E-04	8.352E-04	3.560E-04	3.075E-04	3.134E-04	5.430E-04	3.075E-04	3.075E-04
	MEAN	3.075E-04	1.030E-03	6.517E-04	3.075E-04	1.045E-03	8.095E-04	4.043E-04	1.466E-03
	WORST	3.075E-04	1.789E-03	8.419E-04	3.075E-04	2.036E-03	1.584E-03	1.223E-03	2.036E-02
	STD	1.380E-09	2.604E-04	1.240E-04	2.002E-16	5.664E-04	2.716E-04	2.809E-04	4.465E-03
	H	1	0	1	1	1	0	1	1
F16	BEST	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00
	MEAN	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00
	WORST	-1.032E+00	-1.032E+00	-1.032E+00	-1.032E+00	-1.031E+00	-1.032E+00	-1.032E+00	-1.032E+00
	STD	6.234E-13	2.278E-16	4.727E-09	1.765E-16	5.773E-05	3.688E-10	2.100E-16	2.278E-16
	H	1	0	0	0	0	0	1	1
F17	BEST	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.979E-01
	MEAN	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.980E-01	3.979E-01	3.979E-01	3.979E-01
	WORST	3.979E-01	3.979E-01	3.979E-01	3.979E-01	3.985E-01	3.979E-01	3.979E-01	3.979E-01
	STD	8.509E-09	0.000E+00	6.750E-08	3.209E-13	1.412E-04	9.324E-10	0.000E+00	0.000E+00
	H	0	0	1	0	1	1	0	0
F18	BEST	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.000E+00
	MEAN	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.008E+00	3.000E+00	3.000E+00	3.000E+00
	WORST	3.000E+00	3.000E+00	3.000E+00	3.000E+00	3.024E+00	3.000E+00	3.000E+00	3.000E+00
	STD	2.955E-15	9.153E-12	3.308E-06	1.111E-15	6.744E-03	6.480E-09	1.166E-15	5.094E-16
	H	0	0	1	1	1	0	0	0
F19	BEST	-3.863E+00	-3.863E+00	-3.863E+00	-3.863E+00	-3.862E+00	-3.863E+00	-3.863E+00	-3.863E+00
	MEAN	-3.863E+00	-3.863E+00	-3.863E+00	-3.863E+00	-3.858E+00	-3.863E+00	-3.863E+00	-3.863E+00
	WORST	-3.863E+00	-3.863E+00	-3.863E+00	-3.863E+00	-3.850E+00	-3.863E+00	-3.863E+00	-3.863E+00
	STD	1.750E-15	2.278E-15	7.571E-13	2.152E-15	3.629E-03	3.394E-10	2.258E-15	2.278E-15
	H	0	0	1	0	1	0	1	0

Table E4
Fixed dimensional Multi-modal benchmarks results F20–F23

	CV19T	ABC	ACO	CHIO	COOT	COVIDOA	GA	ICA	PSO
F20	BEST	-3.322E+00	-3.322E+00	-3.322E+00	-3.322E+00	-3.247E+00	-3.322E+00	-3.322E+00	-3.322E+00
	MEAN	-3.239E+00	-3.310E+00	-3.322E+00	-3.316E+00	-3.119E+00	-3.292E+00	-3.322E+00	-3.280E+00
	WORST	-3.203E+00	-3.203E+00	-3.322E+00	-3.203E+00	-3.016E+00	-3.203E+00	-3.322E+00	-3.203E+00
	STD	5.590E-02	3.659E-02	2.228E-08	2.659E-02	7.040E-02	5.282E-02	4.441E-16	5.818E-02
	H	\	0	1	1	1	1	1	1
F21	BEST	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01
	MEAN	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-6.888E+00	-5.655E+00	-1.015E+01	-5.894E+00
	WORST	-1.015E+01	-1.015E+01	-1.015E+01	-1.015E+01	-2.683E+00	-2.630E+00	-1.015E+01	-2.630E+00
	STD	1.823E-15	5.603E-15	3.710E+00	1.025E-04	3.155E+00	3.491E+00	2.305E-15	3.353E+00
	H	\	0	1	1	0	1	0	0
F22	BEST	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01
	MEAN	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-9.609E+00	-7.323E+00	-9.610E+00	-8.110E+00
	WORST	-1.040E+01	-1.040E+01	-1.040E+01	-1.040E+01	-5.087E+00	-2.752E+00	-5.088E+00	-2.752E+00
	STD	4.735E-15	5.345E-15	2.685E+00	1.188E-04	1.937E+00	3.527E+00	1.937E+00	3.594E+00
	H	\	1	1	1	1	1	1	1
F23	BEST	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01
	MEAN	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-7.888E+00	-8.471E+00	-9.596E+00	-7.520E+00
	WORST	-1.054E+01	-1.054E+01	-1.054E+01	-1.054E+01	-2.871E+00	-2.427E+00	-3.835E+00	-2.427E+00
	STD	5.420E-05	8.252E-15	2.919E+00	2.626E-04	3.073E+00	3.303E+00	2.310E+00	3.840E+00
	H	\	1	1	1	1	1	1	1

Table E5
Results of real-life engineering problems

	CV19T	ABC	ACO	CHIO	COOT	COVIDOA	GA	ICA	PSO
TBDT	BEST	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02
	MEAN	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02
	WORST	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.864E+02	1.865E+02	1.864E+02	1.864E+02
	STD	8.477E-14	1.485E-06	2.169E-03	6.745E-14	4.641E-03	2.190E-02	1.075E-13	5.257E-14
	H	\	1	0	1	1	1	1	1
SCBD	BEST	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19
	MEAN	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19
	WORST	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19	9.477E+19
	STD	1.638E+04	1.681E+04	1.595E+04	1.681E+04	1.638E+04	2.813E+04	1.681E+04	2.024E+04
	H	\	0	1	0	0	0	0	0
MDCBD	BEST	1.010E+07	2.074E+14	2.074E+14	2.074E+14	2.798E+14	2.074E+14	2.250E+07	2.074E+14
	MEAN	1.010E+07	2.074E+14	2.074E+14	2.074E+14	4.401E+14	2.074E+14	2.264E+07	2.074E+14
	WORST	1.010E+07	2.074E+14	2.074E+14	2.074E+14	8.038E+14	2.074E+14	2.276E+07	2.074E+14
	STD	0.000E+00	6.412E-02	6.412E-02	6.412E-02	1.296E+14	6.412E-02	1.019E+05	6.412E-02
	H	\	1	1	1	1	1	1	1
HTBD	BEST	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.363E+11	2.365E+11
	MEAN	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.365E+11	6.471E+11	1.307E+12	3.605E+11
	WORST	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.365E+11	2.759E+12	2.736E+12	2.716E+12
	STD	6.262E-05	6.262E-05	6.262E-05	6.262E-05	6.262E-05	8.673E+11	1.215E+12	5.545E+11
	H	\	0	1	0	0	0	0	0

F. Results curves

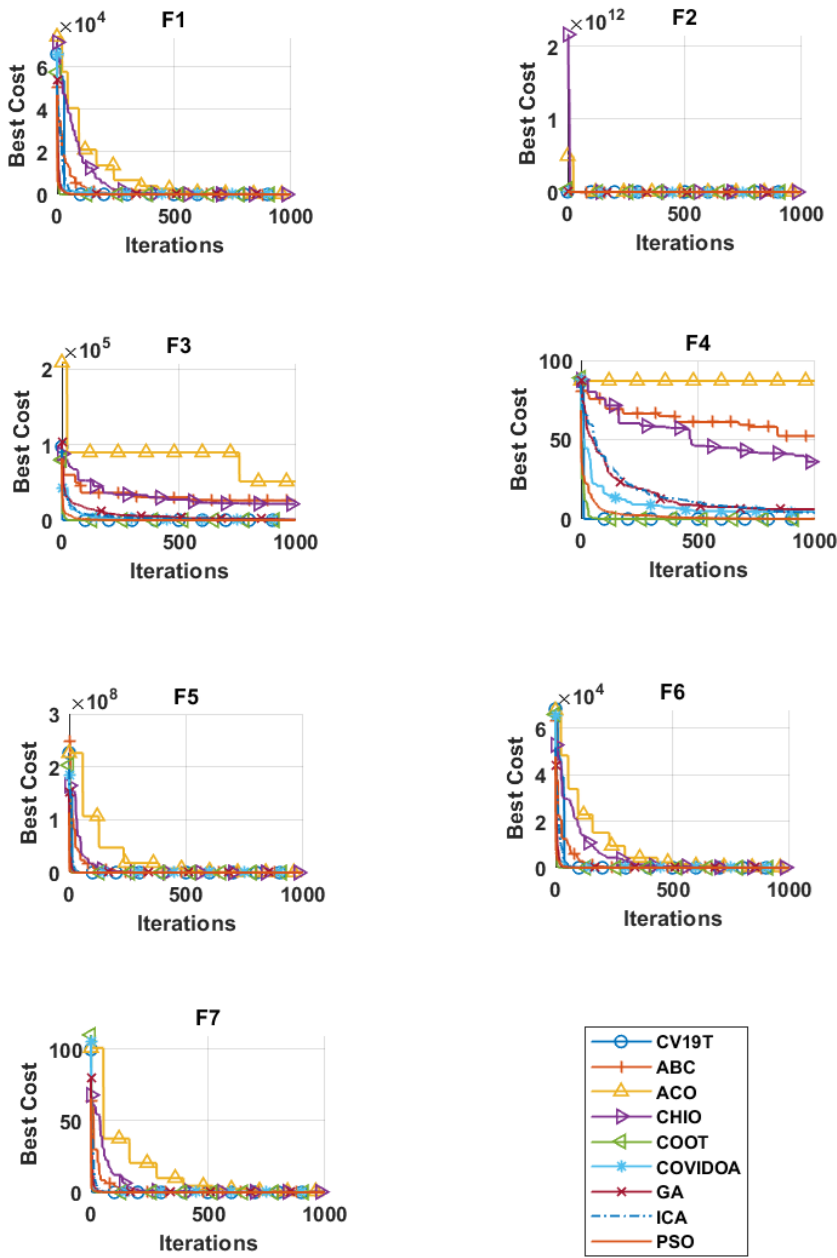


Figure F1. Uni-Modal benchmark functions convergence curves

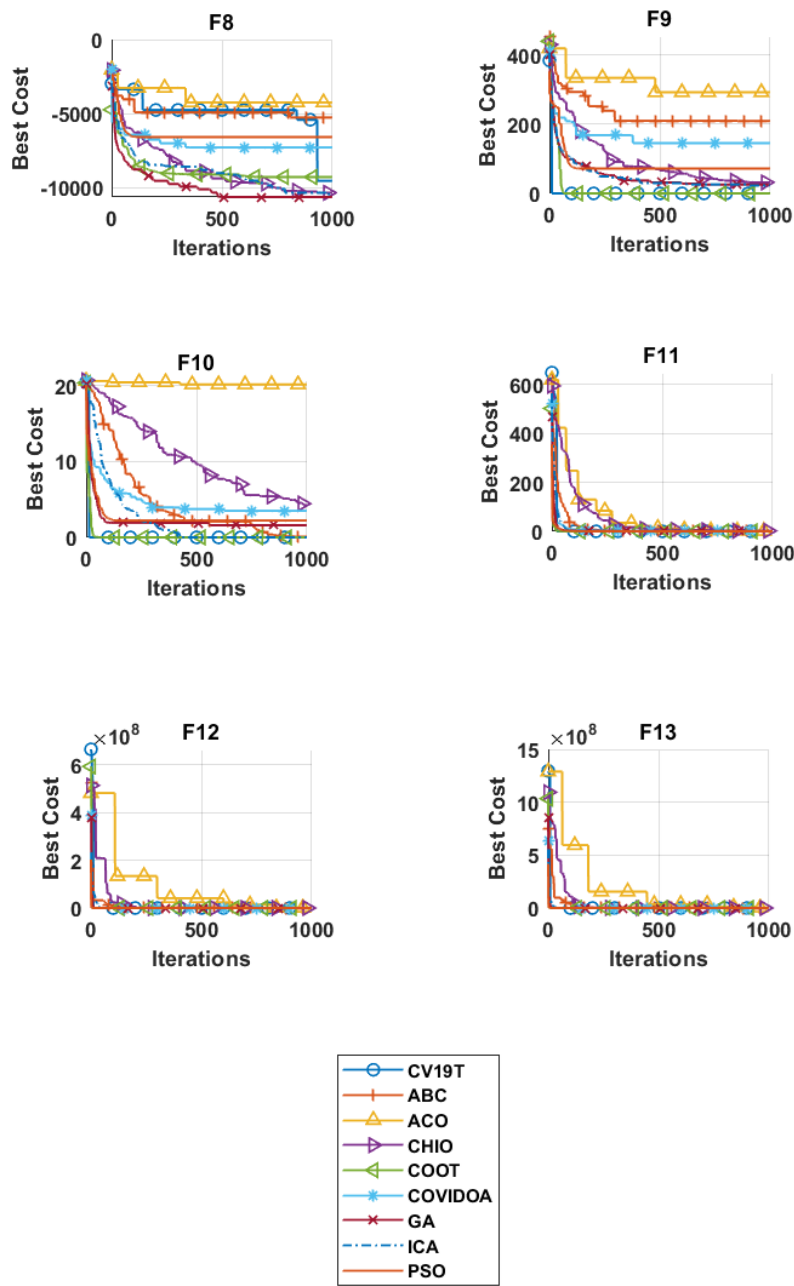


Figure F2. Multi-modal benchmark functions convergence curves

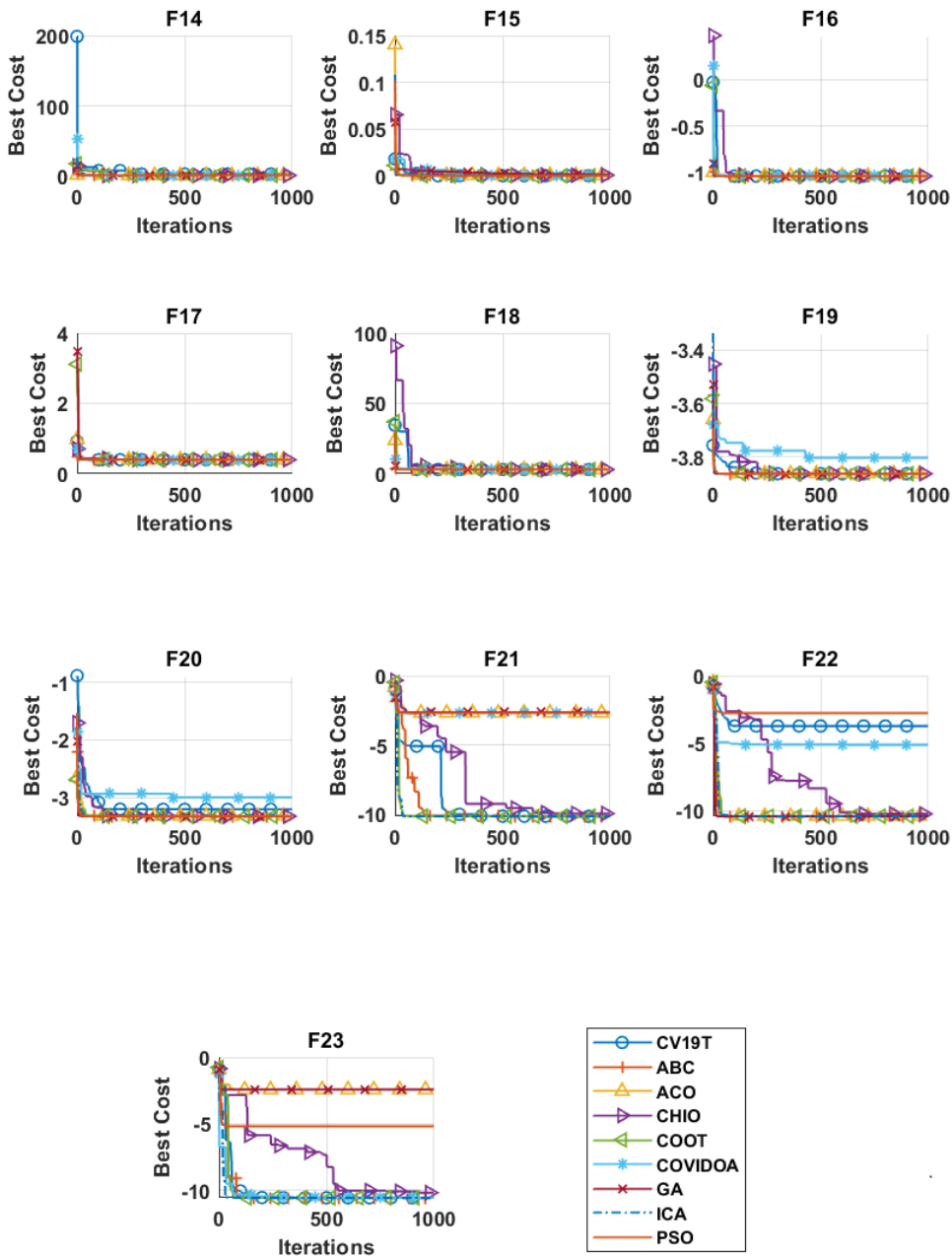


Figure F3. Fixed-dimension multimodal benchmark functions convergence curves

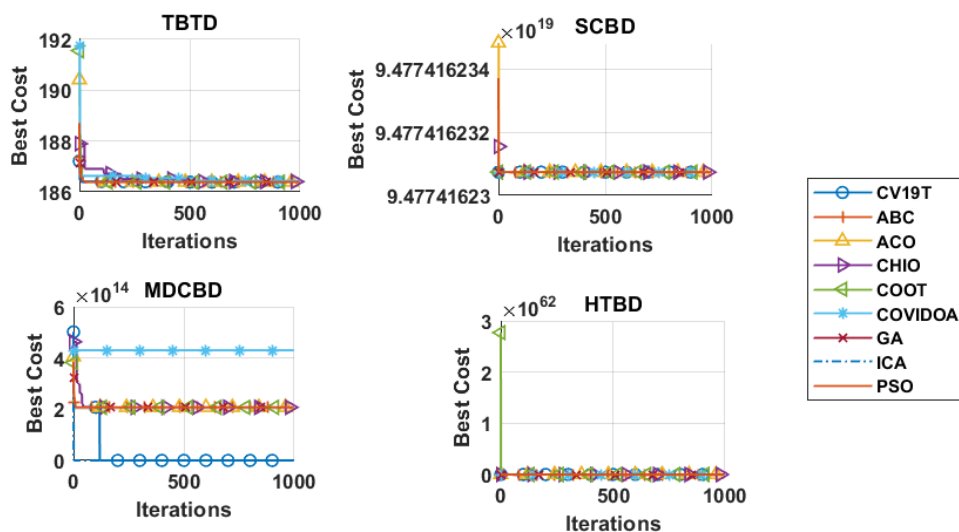


Figure F4. Real-life engineering problems functions convergence curves

Affiliations

Saib Bouthina

Mustapha Ben Boulaid University, Computer Science Departement, Mathematics and Computer Science Faculty, Batna, Algeria, b.saib@univ-batna2.dz

Mohamed-Rida Abdessemed

Mustapha Ben Boulaid University, LAMIE laboratory, Computer Science Departement, Mathematics and Computer Science Faculty, Batna, Algeria, r.abdessemed@univ-batna2.dz

Riadh Hocine

Mustapha Ben Boulaid University, Computer Science Departement, Mathematics and Computer Science Faculty, Batna, Algeria, riadh.hocine@univ-batna2.dz

Received: 6.10.2023

Revised: 25.03.2024

Accepted: 25.03.2024

CHAFIKA DJAOUI
ALLAOUA CHAOUI

FORMALIZATION AND ANALYSIS OF UML 2.0 INTERACTION OVERVIEW DIAGRAM USING MAUDE REWRITING LOGIC LANGUAGE

Abstract *The visual modeling language UML embodies object-oriented design principles. It provides a standard way to visualize the design of a system. It exploits a rich set of well-defined graphical notations for creating abstract models. However, the power of UML is lessened through partially specified formal semantics. Indeed, UML notations are semi-formal and do not lead to fully formalized and executable semantics. Fortunately, UML diagrams are prone to early formalization. Formal methods are a valuable tool that can help overcome the UML constructs' shortage of firm semantics. It is a powerful way to ascribe precise semantics to the graphical notations used in UML diagrams and models. Our work aims to support the semantics of the UML Interaction Overview Diagram. It introduces an approach to leveraging the strengths of the Maude Rewriting Logic language as a formal specification language. The proposal relies on a model-driven engineering approach. It aims to automate the UML Interaction Overview Diagram's mapping to a Maude language specification. The Maude language and its linked tools, including the Maude Model Checker, are used to analyze and verify the resulting Maude specification. Finally, an application example shows the feasibility and benefits of the proposed approach.*

Keywords UML Interaction Overview Diagrams, formal methods, rewriting logic, Maude language, Model-Driven Engineering (MDE), EMF, Sirius, Acceleo

Citation Computer Science 25(3) 2024: 397–419

Copyright © 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

Since its outset, the Object Management Group UML standard has become a general-purpose modeling language and a vital component of the Object-Oriented systems industry [31,32]. It has considerably influenced the specification and development of object-oriented software systems. UML offers a standardized and visual approach to portraying the design and architecture of software systems. It encompasses a variety of simple graphical and textual representation techniques in different phases of the system development cycle. These representation techniques allow the construction of abstract representations (called models), which facilitate the development process and reduce the complexity of the produced system [25].

UML is a semi-formal language rich in syntax and imprecise in semantics. The software developers cannot deny the significant gap between syntax and semantics in the UML-built models. Although the semi-formal nature of UML is an upbeat factor for its convenience and practicality [6], the correctness checking of models requires a firm semantics foundation. The solid semantics foundation caters to a rigorous study of UML diagrams and provides improved accuracy in reasoning about their properties. Nowadays, systems are getting more complicated and need ways to predict problems early in the development process. Anticipating fault detection leads to a successful system specification with low cost and high quality. Hence, it is advisable to map the semi-formal UML notations to concise notations to inspect efficiently whether a model meets the designer's intentions [7]. Formal methods and mathematical techniques are the more powerful approaches that lead to efficient ways to enhance the analytical capabilities of the UML notations.

The OMG has supervised a massive revision of UML 1.X [12], to enlarge the language expressiveness and relevance [34]. Indeed, UML has been revised noticeably through its successive versions (it has now reached version 2.5). UML Version 2.0 has improved the control flow views [18] through a new variant of interaction diagrams called Interaction Overview Diagram (IOD).

An IOD is a two-level behavioral diagram that covers the overall concepts and notations of the Activity Diagram (AD). It highlights interactions within the system and illustrates the control flow at a high level of abstraction. It is a variant of the AD that entails various components, such as interactions, control nodes, and decision points at the top level, also known as the overview level. However, the interaction level (lower level) incorporates detailed interaction diagrams that enable more explicit venturing of specific interactions when needed. An IOD serves as a workflow, business process viewer, or use case that requires more than one interaction diagram to represent multiple flows within a system. Therefore, an IOD is useful for decomposing and modeling complex scenarios that entail the representation of many alternatives in a single diagram. At variance to AD, IOD makes explicit which objects or actors perform activities and how they exchange messages with each other. Being a semi-formal language, IOD comes with imprecise semantics that can lead to misinterpretation,

faulty comprehension, and errors. Hence, it is necessary to translate the IOD into a precise specification that can be verified and analyzed through execution.

This work presents an approach to assigning formal semantics to the IOD. It introduces a general semantic framework for formalizing the IOD in a rewriting logic-based formalism. Accordingly, the IOD core constructs are staffed with precise and formal definitions. We exploit Rewriting Logic (RL) and the Maude language as a firm notation that supports the formal specification of a range of modeling languages [26]. Maude and its associated mathematical tools emerge as a three-part benchmark: a declarative programming language, an executable specification language, and a formal verification framework [10]. Indeed, the Maude tool is suitable for formal specification and early model correctness analysis [9]. In this paper, we propose a mapping from the semi-formal IODs notations into a mathematically equivalent specification in the Maude language. The Maude specifications allow for either execution through simulation or verification using the underlying Maude tools. To reach our goal, we propose to use the model-driven engineering (MDE) approach that relies on meta-modeling and model transformations. We employ well-known standards and tools on the Eclipse platform to automate our approach. Our automatic transformation-based approach has the advantage of reducing design errors at the early stage of software development with low cost and time efficiency. Concretely, our work includes the following contributions:

- We propose a simplified EMF meta-model for IODs. Then, we generate a visual modeling environment to edit and manipulate instances of the IOD's meta-model.
- We define Aceleo templates for mapping IOD notations to a corresponding Maude specification, ensuring automatic Maude code generation.
- We implement a Maude-language executable semantic framework. Hence, we validate the correctness of the semantics of the specified IODs by simulation or verification using the Maude model checker

The rest of the paper follows the upcoming sections: Section 2 summarizes an overview of related works. Section 3 addresses the IOD's key concepts. We recall RL and the Maude language in Section 4. The ensuing section outlines the IOD formalization using Maude language as a mathematical framework. In Section 6, we describe the MDE-based approach. Next, we delineate the implementation of the Case study. The last section concludes the paper and gives some perspectives on this work.

2. Related works

UML is widely used as a versatile modeling language for system specification, with a broad scope to represent different domains. However, the UML semantics are informal and unclear, leading to varied interpretations. Strict verification and tool support are only limited to syntactic issues [19]. The previous concerns led the UML formalization research stream to define the hidden semantics under UML diagrams using formal methods that bridge the gap between UML syntax and semantics.

Due to this, many works have attempted to combine the strengths of formal methods and UML flexibility.

In the literature, most of the approaches have shared the idea of transforming UML models into semantic domains that own verification tools such as Petri Nets [4,5]. SPIN and its underlying verification tool PROMELA, are worth citing [23]. Unlike other UML diagrams, the IOD has not been extensively covered. A few research studies dealt with its formalization. To address the semantic deficiency of the IOD authors have exploited the stochastic process algebra PEPA nets in [21]. Whittle [36] has developed a simulation environment for building and evaluating the semantics of IOD structures using hierarchical finite state machines. The research work in [1] has described a method that transformed IOD to Time Petri Net with energy constraints. It has performed an early analysis and validation of the embedded real-time systems' time and energy conditions. Tebibel's studies [3,6] have attributed formal semantics to IOD by a translation to (HCPNs). Besides the IOD control flow semantics formalization, they exploited CPN tools for the resulting HCPNs simulation and analysis. An attempt [24] has been made at formalizing the hierarchical use of IOD, by extending the work presented in [3] using timed CPNs. The approach presented in [2] has taken full advantage of the rich description in combining different interaction nodes into IODs. Some constructs of IODs have been formalized using a temporal logic called TRIO. Further, the authors have verified some user-defined properties by semantics' implementation in the Zot tool. The proposed approach in [11] has provided a Maude language formalization of UML IOD, with elementary interaction nodes. An automatic translation implementing the proposed formalization has been developed in the AToM3 tool [22]. Subsequently, the yielded Maude specification can be validated through simulation using the Maude system.

This paper proposes an automatic approach for formalizing UML IOD models, using the RL Maude language by integrating insights from two previous research works, namely [11] and [20]. The latter proposed a formalization of UML AD in the Maude language. It has treated the control nodes in UML ADs, which are chiefly similar to the overview level in UML IOD since an IOD is a variant of an AD. At the interaction level, our work revises and extends work [20] by introducing multiple complex interaction scenarios with nested combined fragments (CFs). Indeed, CFs model concurrent behaviors that complicate the analysis of the interaction nodes. Through this work, we intend to translate the IOD into a unified logical and semantic framework. That framework allows a rigorous and well-founded formal analysis to ensure the correctness and reliability of UML-based software systems.

Compared to other approaches, the primary advantage of our work lies in the versatility and universality of the Maude mathematical notations. Maude has good representational capabilities, which allow the integration of all concepts and notions of a language in a single semantic framework. Indeed, Maude does not use any linguistic construction that warps and hides the specific characteristics of each language or domain. "Everything in the Maude specifications is a direct definition of the

language” [13]. A Maude specification is executable. Therefore, Maude gives a formal executable specification to non-executable semi-formal UML models. Moreover, Maude boasts a highly extensible software infrastructure that functions as a mathematical environment. Within this framework, users can leverage a plethora of analysis techniques. Specifically simulation, model checking of invariants, and linear temporal logic (LTL) model checking, to effectively validate or verify properties inherent in Maude executable specifications.

3. Interaction overview diagram constructs

An IOD is a behavioral diagram in UML 2.0. It gives a complete, high-level abstract view of a system under development by showing the flow of interactions between its components. It is a two-level diagram that strikes a balance in providing a broad understanding of the system’s control flow and enabling in-depth exploration of specific interactions when required. It is adequate for yielding an overview of the flow of control, including synchronization, conditional branching, and activity concurrency. Further, it represents the interactions between different objects/actors in the system and pictures the overall dynamics of the system in a single diagram. Being a two-level diagram, an IOD helps designers manage the complexity of broad systems by decomposing the information into modular and understandable components. Thus, identifying potential issues early in the design process makes it easier to change the system to meet the requirements and save time and effort. An IOD shares the general structure of an AD to model the overview level of the system flow and describes interactions within the system using interaction diagram nodes that illustrate the invoked activities or operations.

Syntactically, see Figure 1, an IOD is a connected graph that uses UML AD control constructs (Initial, Decision, Merge, Fork, Join, and Final nodes) to illustrate control flow at the overview level. Instead of Activity elements, it uses rectangular elements to represent interaction nodes that display UML interaction diagrams. An interaction node can be any diagram of the four interaction diagram types (Sequence, Timing, Communication, or another interaction diagram). In this work, we are interested in IODs where interaction nodes are UML Sequence Diagrams (SDs). The utilization of SDs as interaction nodes simplifies the design process and focuses on a sequence of execution for different interactions in the system [29]. Interactions comprise, at the Interaction level, a set of lifelines. Each lifeline symbolizes a role attributed to an object/actor associated with a pertinent class in the system. Interactions over time are depicted as methodical, top-down-arranged messages, visually represented by arrows leading from the source lifeline to the target lifeline

Further, interactions can be regrouped compactly and concisely with Combined Fragments (CFs). CFs (loop, alternative, Option, etc.) define multiple types of control flows using an operator and one or more interaction operands.

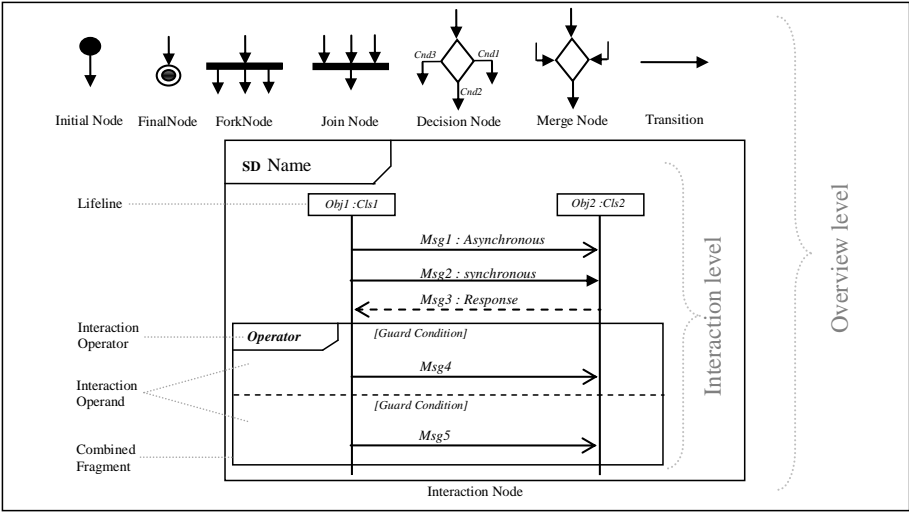


Figure 1. Interaction Overview Diagrams Constructs

The interaction operator shows the logic of the fragment. It describes the semantics of the CF and determines how to use its interaction operands. An interaction operand is a container that groups the interaction fragments and messages exchanged, assuming the guard condition(s). Table 1 provides information on some types of operators and their corresponding descriptions.

Table 1
Some Combined Fragments Operators

Operators	Description
Alt	Alternative: It represents a choice of behavior (among several operands)
Opt	Option: It represents the choice of behavior. It has only one operand to be selected if the guard condition is evaluated to true. Otherwise, the execution flow skips the behavior
Break	It represents a breaking scenario. It has one operand with guard condition. If the guard is true, the operand is chosen, and the rest of the enclosing interaction is skipped. Otherwise, the Break operand is ignored, and the rest of the enclosing interaction is selected
Par	Parallel: It represents the parallel execution of behaviors described within its operands
Loop	It represents an iterated execution of behavior a defined number of times or when a guard condition is evaluated as false
Strict	Strict Sequencing: It defines the strict ordering of the operands
Neg	Negative: It represents a set of negative traces that occur when the system has failed

4. Rewriting logic & Maude language

This section presents Rewriting Logic (RL) and the underlying Maude language. Since its introduction as a unifying framework for concurrency, RL has proven its completeness and suitability as an ideal logic for the specification and analysis of concurrent systems, where concurrent computation is precisely represented by logical deduction [27]. It is an executable semantic framework where different computational models can be specified and executed. More precisely, it is excellent support for giving precise semantics to a variety of concurrent model notations by assigning fully defined formal executable specifications.

A specification in RL is named rewriting theory. Essentially, a rewriting theory includes a signature (which makes up an equational theory) and a set of labeled (conditional) rewriting rules. The signature characterizes the static structure of the system (data structures and operations on them), whereas rewriting rules model the dynamic of the system features [30]. Based on rewriting logic, Maude is a mathematically well-founded declarative specification and programming language where the basic units of specification and programming are theories in RL [28]. The Maude system supports algebraic specification execution and formal analysis technique application. A Maude program (called a Maude module) is precisely a rewriting theory satisfying elementary execution conditions. The calculation in Maude involves logical inference through rewriting. Maude's modules are user-definable and are of two types:

- The functional module defines the system's static aspect as an equational theory. It is a specification in membership equational logic, where data types are declared and operations that act upon them are defined in a precise and rigorous way.
- system module specifies the behavior of a system. It models the dynamics of concurrent systems as a set of rewriting rules that describe local transitions between states [8].

Besides its expressiveness, Maude provides formal reasoning capabilities, including automatic validation and verification tools. For system specification verification, Maude supports the model-checking technique. Nowadays, it is the most popular technique used to prove that a system has no faults and meets its specifications. Maude's model checker is a Linear Temporal Logic properties (LTL properties) verifier. Under the right conditions, the Maude model checker tool can verify LTL properties over a logical finite state space [17]. The result of the verification can either be:

- A fix point (a finite state space) is reached, and the formula is fully verified.
- Conversely, an actual counter-example is offered, proving the violation of the property in question.

5. Formalization of IOD diagrams using Maude language

This section showcases the UML IOD formalization using Rewriting Logic and Maude. By formalizing, we intend to permit the verification and validation of UML diagrams based on the analysis results obtained from the equivalent Maude specifications. To

formalize the UML IOD using Maude language, we have first defined a Basic.IOD functional module that represents the static aspect of UML IOD described using basic types and operations on them. This module is shown in Figure 2.

```
fmod BASIC-IOD is
including STRING .
sort IOD-Config .
op none : -> IOD-Config(ctor) .
op _ : IOD-Config IOD-Config -> IOD-Config(assoccomm id: none) .
**** sort of activity chart : the overview level
Sorts NodeName NodeType IntractionName Interaction .
ops Initial Final : ->NodeName .
ops InitialNode FinalNode : ->NodeType .
op<_> : NodeName NodeType -> IOD-Config .
op[_] : IntractionName Interaction -> IOD-Config .
ops Start End : -> Interaction .
**** Definition sort and operation of Interaction in SD Node
*** object definition
sorts Object Oid ObjType .
ops Actor Obj : ->ObjType .
op _ : Oid ObjType -> Object .
*** message definition
Sorts Msg MsgId MsgType .
Ops Syn Asyn Rep : ->MsgType .
op _ : MsgId MsgType ->Msg .
*** exchanged and sanning Message :an interaction between objects.
Sort MsgSending .
Subsort MsgSending<Interaction .
op<_> : Object Msg Object ->MsgSending .
*** Combined Fragment definition
Sorts CombFragment OperdName Condition .
Subsort CombFragment<Interaction .
subsort String <Condition
op Par[_] : OperdName MsgSending ->CombFragment .
op Par[_] : OperdName CombFragment ->CombFragment .
op Opt[_] : Condition MsgSending ->CombFragment .
op Opt[_] : Condition CombFragment ->CombFragment .
op Alt[_] : Condition MsgSending ->CombFragment .
op Alt[_] : Condition CombFragment ->CombFragment .

**** Loop definition
sort LoopStatus .
ops Entry Exist : ->LoopStatus .
op Loop{_,_} : Nat Nat Condition LoopStatus ->CombFragment .
op Loop{_,_} : Nat Nat Condition MsgSending ->CombFragment .
op Loop{_,_} : Nat Nat Condition CombFragment ->CombFragment .

endfm
```

Figure 2. Basic Interaction Overview Diagram Functional Module

In the Basic.IOD module, we define a new type called ‘IOD_Config’ that represents the current nodes of an IOD diagram instance (execution occurrence). The ‘none’ constant denotes the empty configuration in an IOD. In our approach, current nodes are the Initial node, Final node or interaction node. To specify the initial node and final node, we define the operation “< _ : _ >”. The first parameter of this operation is a constant of the ‘NodeName’ sort which represents the name of the node, whereas the second one is used to specify the type of the node, which can be ‘initial Node’ or ‘final Node’.

The interaction nodes in IOD are defined by the operation “[_|-]”, where the first parameter of this operation is the name of the interaction node, whilst the second one is used to represent an interaction inside the node (message passing between objects). We have also defined in the second parameter two flag values (constants) denoting the beginning of the invocation (“Start”) and the end of the invocation (“End”) of the interaction node.

The control flow between interaction nodes in the overview level is formalized using rewriting rules. Rewriting rules represent transitions firing or controlling nodes with their conditions. We note the proposed overview level formalization is adapted from [20] that formalizes UML AD using Maude language. Table 2 summarizes the rewriting rules corresponding to the principal structures of the overview level.

Interaction nodes in the interaction level of the IOD emphasize object interactions. An interaction node contains lifelines (object/Actor) and exchanged messages to represent a single scenario as a smaller SD.

To describe the sending of messages between objects in Maude, we have created two general sorts, called (“Object” and “Msg”). The first one is defined using the operation “_:_” where the first parameter is a constant of the “Oid” sort that represents the object name, whereas the second one is used to specify the object type. The “Msg” sort displays a message defined by the operation “_:_” that denotes the “id” of the message in the first parameter and the type in the former. An exchanged message (a simple interaction) is outlined with the operation “< _, _, _ >” (defined in “MsgSending” sort) where the first parameter is the sender object, the second is the message sent, and the last one is the receiver object. Since sent messages are attached to the name of the interaction node where they participate, “MsgSending” types are declared as sub-sorts of the “Interaction” sort, which is the current interaction.

Furthermore, interactions in IOD diagram can be represented in a compacted form using Combined Fragments (CFs). A CF is defined by an operator which specifies how the operands will be executed. In our approach, we define for each CF operator an operation in Maude (as shown in Basic.IOD module in Figure 2). Table 3 depicts the rewriting rules corresponding to each operator.

We note that these rules are valid even for interactions between three objects and more, and the proposed formalization allows the nesting of the CFs.

Table 2
Mapping Control Structures to Maude


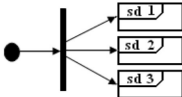
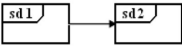

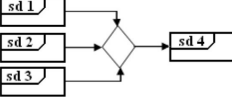
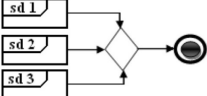
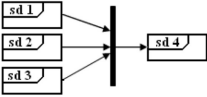
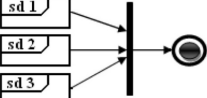
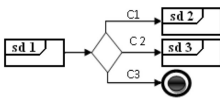
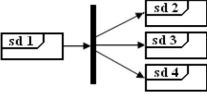
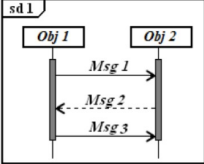
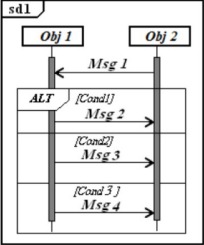
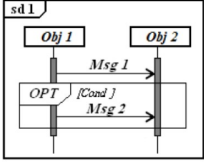
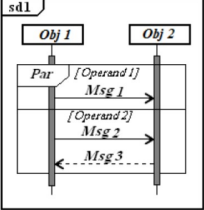
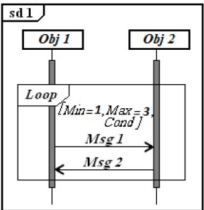
Interaction Overview Diagram Overview Level	Corresponding Maude Rewriting Rules
Initial to SD 	$rl \text{ [Initial]} : < \text{Initial} : \text{InitialNode} > => [\text{SD1} \text{Start}] .$
Initial to Fork Node 	$rl \text{ [Initial2Frk]} : < \text{Initial} : \text{Initial Node} > => [\text{SD1} \text{Start}] [\text{SD2} \text{Start}] [\text{SD3} \text{Start}] .$
SD to SD 	$rl \text{ [transition]} : [\text{SD1} \text{End}] => [\text{SD2} \text{Start}] .$
SD to Final Node 	$rl \text{ [ToFinal]} : [\text{SD1} \text{End}] => < \text{Final} : \text{FinalNode} > .$
Merge Node 	$rl \text{ [Merge]} : [\text{SD1} \text{End}] => [\text{SD4} \text{Start}] .$ $rl \text{ [Merge]} : [\text{SD2} \text{End}] => [\text{SD4} \text{Start}] .$ $rl \text{ [Merge]} : [\text{SD3} \text{End}] => [\text{SD4} \text{Start}] .$
	$rl \text{ [Merge]} : [\text{SD1} \text{End}] => < \text{Final} : \text{FinalNode} > .$ $rl \text{ [Merge]} : [\text{SD2} \text{End}] => < \text{Final} : \text{FinalNode} > .$ $rl \text{ [Merge]} : [\text{SD3} \text{End}] => < \text{Final} : \text{FinalNode} > .$
Join Node 	$rl \text{ [Join]} : [\text{SD1} \text{End}] [\text{SD2} \text{End}] [\text{SD3} \text{End}] => [\text{SD4} \text{Start}] .$
	$rl \text{ [Join]} : [\text{SD1} \text{End}] [\text{SD2} \text{End}] [\text{SD3} \text{End}] => < \text{Final} : \text{FinalNode} > .$
Decision Node 	$rl \text{ [Decision_C1]} : [\text{SD1} \text{End} \{ \text{Cond} \}] => [\text{SD2} \text{Start}] .$ $rl \text{ [Decision_C2]} : [\text{SD1} \text{End} \{ \text{Cond} \}] => [\text{SD3} \text{Start}] .$ $rl \text{ [Decision_C3]} : [\text{SD1} \text{End} \{ \text{Cond} \}] => < \text{Final} : \text{FinalNode} > .$
Fork Node 	$rl \text{ [Fork]} : [\text{SD1} \text{End}] => [\text{SD2} \text{Start}] [\text{SD3} \text{Start}] [\text{SD4} \text{Start}] .$

Table 3
Mapping Interactions to Maude

IOD lower level	Corresponding Maude Rewriting Rules
<p>Message Passing (Simple Message)</p> 	<pre> rl [SendMsg1] : [SD1 Start] =>[SD1 < Obj1 : Obj, Msg1 : Syn, Obj2: Obj>] . rl [SendMsg2] : [SD1 < Obj1 : Obj, Msg1 : Syn, Obj2: Obj>] => [SD1 < Obj2 : Obj, Msg2 : Ans, Obj1: Obj>] . rl [SendMsg3] : [SD1 < Obj2 : Obj, Msg2 : Ans, Obj1: Obj>] =>[SD1 < Obj1 : Obj, Msg3 : Asy, Obj2: Obj>] . rl [endSD] : [SD1 < Obj1 : Obj, Msg3 : Asy, Obj2: Obj>] =>[SD1 End] . </pre>
<p>Alt Fragment</p> 	<pre> rl [SendMsg1] : [SD1 Start] =>[SD1 < Obj2 : Obj, Msg1 : Asy, Obj1: Obj>] . *** Alt Condition 1 is true rl [AltCond1] : [SD1 < Obj2 : Obj, Msg1 : Asy, Obj1: Obj>] =>[SD1 Alt {Cond1 : < Obj1 : Obj, Msg2 : Asy, Obj2: Obj> }] . rl [AltCond1] : [SD1 Alt {Cond1 : < Obj1 : Obj, Msg2 : Asy, Obj2: Obj> }] =>[SD1 End] . *** Alt Condition 2 is true rl [AltCond2] : [SD1 < Obj2 : Obj, Msg1 : Asy, Obj1: Obj>] =>[SD1 Alt {Cond2 : < Obj1 : Obj, Msg3 : Asy, Obj2: Obj> }] . rl [AltCond2] : [SD1 Alt {Cond2 : < Obj1 : Obj, Msg3 : Asy, Obj2: Obj> }] =>[SD1 End] . *** Alt Condition 3 is true or else statement rl [AltCond3] : [SD1 < Obj2 : Obj, Msg1 : Asy, Obj1: Obj>] =>[SD1 Alt {Cond3 : < Obj1 : Obj, Msg3 : Asy, Obj2: Obj> }] . rl [AltCond3] : [SD1 < Obj1 : Obj, Msg3 : Asy, Obj2: Obj>] =>[SD1 End] . </pre>
<p>Opt Fragment</p> 	<pre> rl [SendMsg1] : [SD1 Start] =>[SD1 < Obj1 : Obj, Msg1 : Asy, Obj2: Obj>] . *** Opt Condition is True rl [CondOptTrue] : [SD1 < Obj1 : Obj, Msg1 : Asy, Obj2: Obj>] => [SD1 Opt {Cond : < Obj1 : Obj, Msg2 : Asy, Obj2: Obj> }] . rl [.....] : [SD1 Opt {Cond : < Obj1 : Obj, Msg2 : Asy, Obj2: Obj> }] =>[SD1 End] . *** Opt Condition is false rl [CondOptFalse] : [SD1 < Obj1 : Obj, Msg1 : Syn, Obj2: Obj>] =>[SD1 End] . </pre>
<p>Par Fragment</p> 	<pre> rl [ParFragment] : [SD1 Start] => [SD1 Par{Operand1 : < Obj1 : Obj, Msg1 : Asy, Obj2: Obj> }] . [SD1 Par{Operand2 : < Obj1 : Obj, Msg2 : Syn, Obj2: Obj> }] . rl [Par_Msg2] : [SD1 Par{Operand2 : < Obj1 : Obj, Msg2 : Syn, Obj2: Obj> }] => [SD1 Par{Operand2 : < Obj2 : Obj, Msg3 : Ans, Obj1: Obj> }] . rl [EndPar] : [SD1 Par{Operand1 : < Obj1 : Obj, Msg1 : Asy, Obj2: Obj> }] [SD1 Par{Operand2 : < Obj2 : Obj, Msg3 : Ans, Obj1: Obj> }] => [SD1 End] . </pre>
<p>Loop Fragment</p> 	<pre> vars count max : Nat . varcond : String . rl [Start Loop] : [SD1 Start] =>[SD1 Loop{1,3,"Cond" : Entry}] . *** Loop execution cr1 [LoopMsg1] : [SD1 Loop{ count, max, cond : Entry}] =>[SD1 Loop{ count, max, cond : < Obj1 : Obj, Msg1 : Asy, Obj2: Obj>}] if (cond == "Cond") /\ (count <= max) . rl [LoopMsg2] : [SD1 Loop{ count, max, cond : < Obj1 : Obj, Msg1 : Asy, Obj2: Obj>}] =>[SD1 Loop{ count, max, cond : < Obj2 : Obj, Msg2 : Asy, Obj1: Obj> }] . *** Loop restart(with "Cond" OR "NotCond") rl [RestartLoopCond] : [SD1 Loop{ count , max, cond : < Obj2 : Obj, Msg2 : Asy, Obj1: Obj>}] =>[SD1 Loop{ count +1, max, "Cond" : Entry }] . rl [RestartLoopNotCond] : [SD1 Loop{ count , max, cond : < Obj2 : Obj, Msg2 : Asy, Obj1: Obj>}] =>[SD1 Loop{ count +1, max, "NotCond" : Entry }] . **** Loop exist cr1 [ExistWithoutLoop] : [SD1 Loop{ count, max, cond : Entry}] =>[SD1 Loop{ count , max, cond : Exist }] if (cond /= "Cond") or (count > max) . rl [EndLoop] : [SD1 Loop{ count, max, cond : Exist}] =>[SD1 End] . </pre>

6. The MDE based approach

In this section, we outline the proposed MDE-based approach to transform the dynamic behaviors of systems expressed using UML IODs into their equivalent Maude specification, by considering the defined formalization in Section 5. The approach comprises a two-step process:

- The first step is to redefine a simplified meta-model for the IOD and build a graphical editor for the diagram according to the proposed meta-model. To fulfill this, we have offered the use of Eclipse Modeling Framework (EMF) [15], which forms the basis for MDE on the Eclipse platform. Based on the proposed IOD meta-model, we have used Sirius framework [16] to build a graphical modeling editor for IOD.
- The second step is to prepare the generation of the equivalent Maude specification of the IOD. For achieving an automatic and correct process of code generation, we have proposed to use the Acceleo language [14] to define and implement the transformations.

In the ensuing section, we unveil in detail our approach.

6.1. IOD Meta-Model

The development of modeling language requires providing both abstract syntax and concrete syntax. Abstract syntax specifies the meta-model constructs, the associated attributes, relationships, and constraints. The concrete syntax determines how the previous constructs are connected and how they visually appear in a graphical editor. In this work, we deal with the formalization and model analysis of a subset of the UML 2.5 meta-model concepts by using a simplified meta-model of the diagram. In Eclipse EMF, a meta-model is established in the Ecore format [33], which is primarily a subset of UML class diagrams. We propose to meta-model the UML IOD with the Ecore model shown in Figure 3. The root node of this meta-model (named `IOD_Diagram`) contains several nodes, namely (`IOD_Nodes`), each modeling either a control node (`IOD_ControlNode`) or an interaction node (`IOD_InteractionNode`). These nodes are connected by edges (`IOD_edges`). These edges can have guards to conditionally branch the control control. There are several types of control nodes for determining the flow direction. This follows the notion of UML AD. `InitialNode`, `FinalNode`, `DecisionNode`, `ForkNode`, `JoinNode` and `MergeNodes` classes describe the control flow at the overview level in the IOD. The interaction nodes are based on the UML SD concepts. An interaction node consists of several interactions among objects/actors. The lifelines (`LifeLine` class) display the interacting objects/actors by exchanging messages (`Message` class). CFs (`CombinedFragment` class) are a composition of interactions defined by an interaction operator (which is specified through the `Type` attribute), and embedded in corresponding interaction operands (the `Operator` class). An operand can have interaction constraints (such as guards or loop parameters for a loop operator).

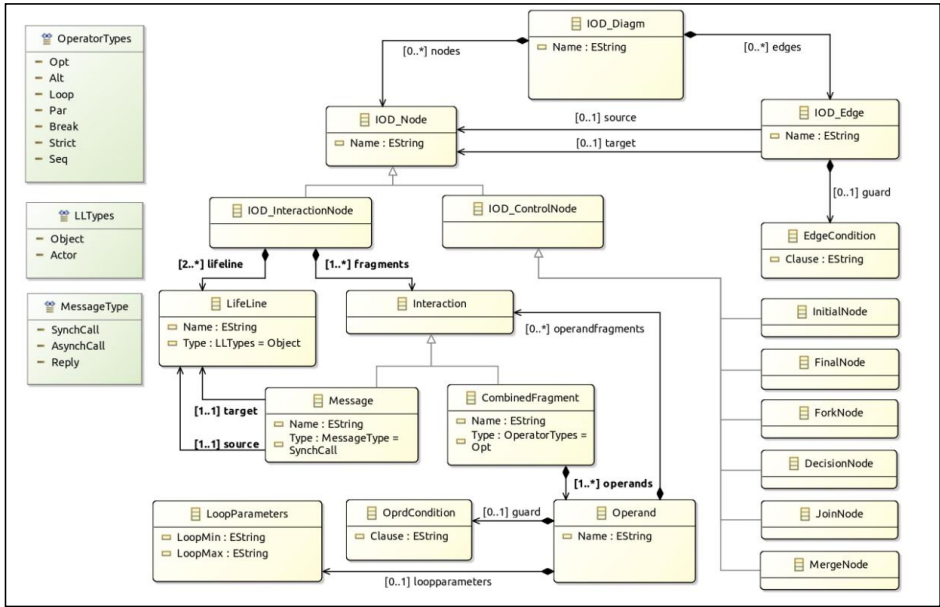


Figure 3. Interaction Overview Diagram Meta-Model

Using the proposed Ecore model, we can generate a user-friendly tree-based editor for the UML IOD modeling language. This editor allows easy editing and viewing of model instances through EMF. To develop its graphical modeling editors, we use the Sirius framework. Eclipse, EMF, and Sirius technology grant the creation of customized graphical modeling environments. The users can create, visualize, and update IOD models by using the custom modeling environment. (see the palette toolbar in Figure 5).

6.2. Maude Code generation

The next step is to transform the UML IOD specification into its equivalent Maude specification. According to the presented formalization in Section 5. The transformation is performed by using the Acceleo transformation language. Acceleo is a template-based technology. It includes tools for creating custom code generators that allow the automatic generation of any code from a data source available in an EMF format. Acceleo models faithfully translate the IOD formalization into corresponding constructs in the Maude language. To establish Maude's specification, we have defined a set of atomic mapping rules from the elementary elements and structures in the source model into elementary constructs in the target model (as defined in Section 5: Figure 2, Table 2, and Table 3). The Maude code generation process is composed of a set of Acceleo templates. Figure 4 shows some templates.

```

[template public IOD2Maude (anIOD_Diagm : IOD_Diagm)]
[comment @main/]
----Maude code generation of the overview level
[file (anIOD_Diagm.Name.concat('.maude'),false,'UTF-8')]
mod [anIOD_Diagm.Name/] is
including BASIC-IOD.
[for (aIOD_InteractionNode :IOD_InteractionNode|anIOD_Diagm.eContents(IOD_InteractionNode))]
including [aIOD_InteractionNode.Name/].
[/for]
Op [anIOD_Diagm.Name/] :-> IOD-Config.
Eq [anIOD_Diagm.Name/] = <Initial:InitialNode>.
--- Maude code generation of control flow between interaction nodes
[aIOD_Diagm.GenControlFlow()/]
endm
[/file]
----Maude code generation of the interaction nodes level
[for (anIOD_InteractionNode :IOD_InteractionNode|anIOD_Diagm.eContents(IOD_InteractionNode))]
[file (anIOD_InteractionNode.Name.concat('.maude'),false,'UTF-8')]
Mod [anIOD_InteractionNode.Name/] is
including BASIC-IOD.
---The objects/actors definition
[aIOD_InteractionNode.GenObject()/]
---The messages definition
[aIOD_InteractionNode.GenMessage()/]
---The CF definition
[aIOD_InteractionNode.GenCombinedFragment()/]
--- Maude code generation of the interactions: messages exchange
op[aIOD_InteractionNode.Name/] :-> IntractionName.
[aIOD_InteractionNode.GenInteractions()/]
endm
[/file]
[/for]
[/template]

[comment : GenControlFlow template: generation of control flow in Maude /]
[template private GenControlFlow (anIOD_Diagm :IOD_Diagm)]
--- code Maude generation of sequential transition
[for (aIOEdge :IOD_Edge|anIOD_Diagm.eAllContents(IOD_Edge)
->select(a|[a.source.ocIsKindOf(InitialNode) and a.target.ocIsKindOf(IOD_InteractionNode)]
or [a.source.ocIsKindOf(IOD_InteractionNode) and a.target.ocIsKindOf(IOD_InteractionNode)]
or [a.source.ocIsKindOf(IOD_InteractionNode) and a.target.ocIsKindOf(FinalNode)]))]
[if aIOEdge.source.ocIsKindOf(InitialNode) and aIOEdge.target.ocIsKindOf(IOD_InteractionNode)]
r1 ['/'/Initial]:<Initial:InitialNode>=>['/'/[aIOEdge.target.Name/]|Start|.
[elseif aIOEdge.source.ocIsKindOf(IOD_InteractionNode) and aIOEdge.target.ocIsKindOf(IOD_InteractionNode)]
r1 ['/'/[aIOEdge.source.Name/]|'-To-
/'/[aIOEdge.target.Name/]|]|End]=>['/'/[aIOEdge.target.Name/]|Start|.
[elseif aIOEdge.source.ocIsKindOf(IOD_InteractionNode) and aIOEdge.target.ocIsKindOf(FinalNode)]
r1 ['/'/ToFinal]:['/'/[aIOEdge.source.Name/]|End]=><Final:FinalNode>.
[/if]
[/for]
--- code Maude generation of control nodes
[for (aIOD_ControlNode :IOD_ControlNode|anIOD_Diagm.eAllContents(IOD_ControlNode)
->select(a|not(a.ocIsKindOf(InitialNode) or a.ocIsKindOf(FinalNode)))]
[if aIOD_ControlNode.ocIsKindOf(ForkNode)] [aIOD_ControlNode.GenForkNode()/]
[elseif aIOD_ControlNode.ocIsKindOf(MergeNode)] [aIOD_ControlNode.GenMergeNode()/]
[elseif aIOD_ControlNode.ocIsKindOf(DecisionNode)] [aIOD_ControlNode.GenDecisionNode()/]
[elseif aIOD_ControlNode.ocIsKindOf(JoinNode)] [aIOD_ControlNode.GenJoinNode()/]
[/if]
[/for]
[/template]

[comment : GenObject template: generation of Objects/actors definitions in Maude /]
[template private GenObject(anIOD_InteractionNode : IOD_InteractionNode) post(tokenize("\n"))]
[if anIOD_InteractionNode.lifeline<=null]
ops
[for (aName : String|anIOD_InteractionNode.lifeline.Name->asSet())][aName/|] [/for] :-> Oid.
[/if]
[/template]

```

Figure 4. Some Aceleo templates to generate Maude code

The templates browse the source model elements (instance of the UML IOD metamodel) and generate the corresponding Maude code. The main *Acceleo* template is named *IOD2Maude()*. It generates the equivalent Maude code for both levels of the diagram: the overview level and the interaction level. For the overview level, it generates a source Maude code file for the statements of the corresponding Maude modules for each interaction node. After that, the *GenControlFlow()* template generates the Maude code for the control flow between the interaction nodes. The *GenControlFlow()* template first generates the equivalent rewriting rule for each sequential transition firing. Afterward, according to the defined semantics, the appropriate template is used to generate the corresponding rewriting rule(s) for each type of control node. The main template generates a text file for each interaction node at the interaction level. The text file saves the resulting Maude code from the mapping of the underlined interaction objects, messages, CFs and interactions by applying different templates. For example, the *GenObjet()* template is applied for the generation of the node-underlined object/actor Maude specification.

7. Case study

The efficiency of our approach is assessed through an online virtual bookstore [35] example. A virtual bookstore is a digital framework that allows customers to purchase books online. The bookstore typically has a wide selection of accessible books for sale, which can be easily searched and filtered based on various criteria. Customers can access the online bookstore through a front-end-user interface. They can enter keywords to search for specific titles or authors, browse the books listed in the library, view book details, and order books. To order books, customers can add books to their shopping cart, view their orders, and make secure online payments. Figure 5 shows an IOD that depicts the behavioral aspect of the online bookstore system. The IOD portrays essential nodes, namely: initial, final, decision, merge, and four interaction nodes (*OrderBooks*, *ConnectToSystem*, *RegisterOrder*, and *OrderPayment*). Each interaction node includes a UML SD that involves three actors: the customer, the lib interface (user interface), and the lib system control. The latter monitors the overall operations of the online bookstore system. It carries out operations, including inventory management and payment processing for books.

The diagram starts at the initial node and leads to the first interaction node that models the book order. In this node, the system iterates over each keyword in the search query and retrieves a list of matching books. Once the system has retrieved a list of books for each keyword in the search query, it creates a new list of books that fits all the keywords in the query. The last list is added to a search results set and displayed to the user. Customers can cancel the order at any point before completing the checkout process. In this scenario, the diagram concludes at the final node. Before finalizing the order, the online bookstore invites the customer to log into his account if he is not already logged in. For a new customer, the online bookstore asks him to create a new account. Once a customer has connected to his account, he can complete

the order. He will confirm the details of his order in the RegisterOrder interaction node by entering his shipping and billing information. Then, in the OrderPayment interaction node, he makes the payment using a secure online payment system. The OrderPayment interaction node introduces a new actor named CreditCardOperator. This actor manages the responsibility of transaction payments between the bookstore and the customer.

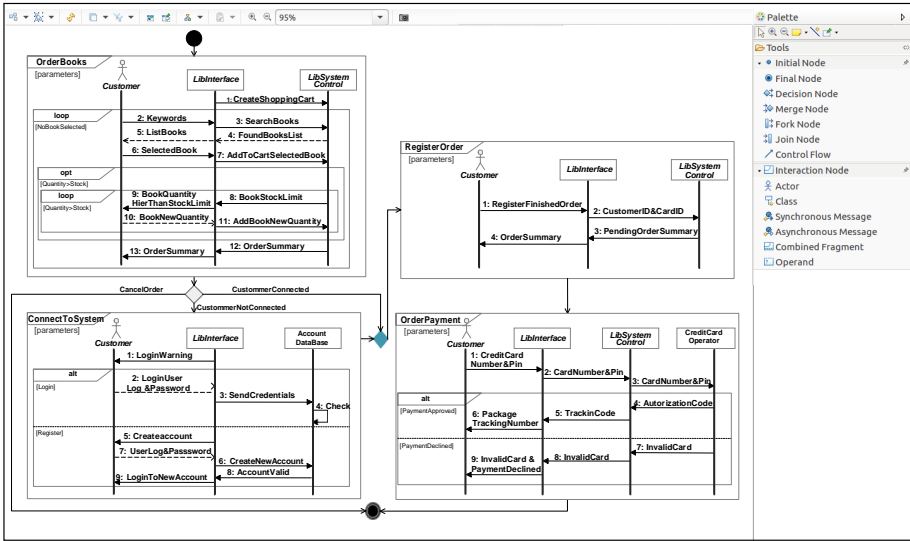


Figure 5. Specification of the OnlineBookStore IOD

7.1. Generation of Maude specification

To thoroughly assess the virtual bookstore model, it is imperative to map it into its equivalent Maude specification. The bookstore model Maude formalization follows patterns and instructions specified in the Aceleo templates. Aceleo templates define how to navigate the elements of the bookstore as the EMF input model and what code to generate. After the Aceleo template execution, the corresponding Maude code to the model of the virtual bookstore is generated. We have used Maude's modularity feature to generate separate Maude modules for the overview level and each interaction node. Figure 6 shows the Maude system module of the OnlineBookStore IOD. It outlines the Maude specification of the virtual bookstore model.

This module includes the Basic.IOD functional module, where the static aspect of the diagram (actors, messages, CFs, interactions) is declared using different types and operators. Each interaction node's code is also included by calling its corresponding Maude module. In addition, this module defines the operator OnlineBookStore as IOD-Config sort. It corresponds to the name of the IOD model. It also introduces the OnlineBookStore equation to indicate the first state of the IOD's execution. The

module ends with the equivalent rewriting rule(s) for each sequential transition firing or control node.

```

mod OnlineBookStore is
including BASIC-IOD.
including OrderBooks .
including ConnectToSystem .
including RegisterOrder .
including OrderPayment .
op OnlineBookStore : -> IOD-Config .
eq OnlineBookStore = < Initial : InitialNode > .
--- The control flow between interactions
rl [Initial] : < Initial : InitialNode > => [ OrderBooks | Start ] .
rl [Decision-CancelOrder] : [ OrderBooks | End ] => < Final : FinalNode > .
rl [Decision-CustomerNotConnected] : [ OrderBooks | End ] => [ ConnectToSystem | Start ] .
rl [Decision-CustomerConnected] : [ OrderBooks | End ] => [ RegisterOrder | Start ] .
rl [ConnectToSystem-To-RegisterOrder] : [ConnectToSystem | End ] => [ RegisterOrder | Start ] .
rl [RegisterOrder-To-OrderPayment] : [RegisterOrder | End ] => [OrderPayment | Start ] .
rl [ToFinal] : [OrderPayment | End ] => < Final : FinalNode > .
endm

```

Figure 6. The Generated OnlineBookStore System Module: the overview level of IOD

Each interaction node is depicted in a different system module. For example, The RegisterOrder system module, shown in Figure 7, contains the rewriting rules specifying interactions (exchanging messages) between different objects in the RegisterOrder interaction node.

```

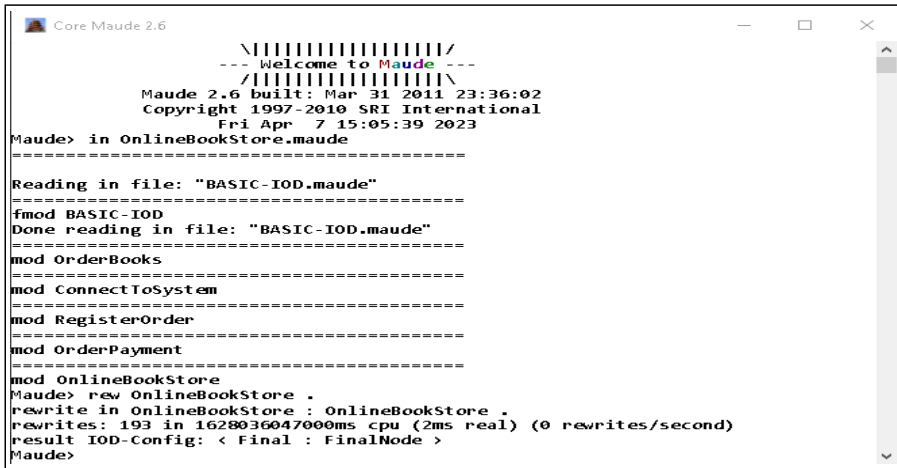
mod RegisterOrder is
including BASIC-IOD .
--- The objects
ops Customer LibInterface LibSystemControl : -> Oid .
--- The OrderSummary messages
ops RegisterFinishedOrder CustomerID&CardID PendingOrderSummary OrderSummary : -> MsgId .
--- The interaction
op RegisterOrder : -> IntractionName .
rl [RegisterFinishedOrder] : [ RegisterOrder | Start ]
=> [ RegisterOrder | < Customer : Skm , RegisterFinishedOrder : Syn , LibInterface : Obj > ] .
rl [CustomerID&CardID] : [ RegisterOrder | < Customer : Skm , RegisterFinishedOrder : Syn , LibInterface : Obj > ]
=> [ RegisterOrder | < LibInterface : Obj , CustomerID&CardID : Syn , LibSystemControl : Obj > ] .
rl [PendingOrderSummary] : [ RegisterOrder | < LibInterface : Obj , CustomerID&CardID : Syn , LibSystemControl : Obj > ]
=> [ RegisterOrder | < LibSystemControl : Obj , PendingOrderSummary : Syn , LibInterface : Obj > ] .
rl [OrderSummary] : [ RegisterOrder | < LibSystemControl : Obj , PendingOrderSummary : Syn , LibInterface : Obj > ]
=> [ RegisterOrder | < LibInterface : Obj , OrderSummary : Syn , Customer : Skm > ] .
rl [RegisterOrderEnd] : [ RegisterOrder | < LibInterface : Obj , OrderSummary : Syn , Customer : Skm > ]
=> [ RegisterOrder | End ] .
endm

```

Figure 7. The Generated RegisterOrder System Module
from the RegisterOrder Interaction Node

7.2. Simulation

We invoked the RL Maude system to perform the resulting Maude specification analysis by simulation. The simulation process involves evolving the system to observe its behavior. In the Maude system, the simulation is composed of iterations where the initial state of the diagram changes by applying one or more rewriting rules. Besides the generated file, the user can limit the number of rewriting steps in the simulator if (s/he) wants to check intermediate states. Otherwise, the simulator continues the simulation operation until reaching a final state for a correct design. The results of the simulation of the library Maude specification are presented in Figure 8. We notice that when the simulator starts from the initial node, the final node is reached.



```

Core Maude 2.6
-----
Welcome to Maude
-----
Maude 2.6 built: Mar 31 2011 23:36:02
Copyright 1997-2010 SRI International
Fri Apr 7 15:05:39 2023
Maude> in OnlineBookStore.maude
-----
Reading in file: "BASIC-IOD.maude"
-----
fmod BASIC-IOD
Done reading in file: "BASIC-IOD.maude"
-----
mod OrderBooks
-----
mod ConnectToSystem
-----
mod RegisterOrder
-----
mod OrderPayment
-----
mod OnlineBookStore
Maude> rew OnlineBookStore .
rewrite in OnlineBookStore : OnlineBookStore .
rewrites: 193 in 1628036047000ms cpu (2ms real) (0 rewrites/second)
result IOD-Config: < Final : FinalNode >
Maude>

```

Figure 8. The Simulation of the Virtual Bookstore IOD within Maude System

7.3. Verification and analysis

This section illustrates the use of the Maude LTL model checker to perform the verification and analyses. LTL is a formalism for expressing system properties using predicates. As Maude supports equational logic, the LTL properties are expressed using equations. To verify a property, we need to define it manually. After its definition, the property is embedded in the specification module, where the system and its behavior are defined. Consider the virtual bookstore specification in the example. A temporal property that the specification must satisfy is termination. The property means that when the system starts from the initial node, it always reaches the final node. We define two predicates (Start and Final) to express the termination property in a new module called IODPREDS. The IODPREDS module encompasses the property along with the bookstore system specification (see Figure 9). The termination property is expressed in LTL as follows:

$$\Box \langle \rangle \text{Final}(\langle \text{Final} : \text{FinalNode} \rangle)$$

```

mod IOD-PREDS is
  protecting OnlineBookStore.
  including SATISFACTION .
  subsort IOD-Config < State .
  op Start : IOD-Config -> Prop .
  op Final : IOD-Config -> Prop .
  var config : IOD-Config .
  eq < Initial : InitialNode > config |= Start (< Initial : InitialNode >) = true .
  eq < Final : FinalNode > config |= Final (< Final : FinalNode >) = true .
endm

```

Figure 9. IOD-PREDS Module

The IODCHECK module verifies this propriety (see Figure 10). In Figure 11, Maude's LTL model checker result shows that the property is verified successfully for the diagram.

```

mod IOD-CHECK is
  protecting IOD-PREDS .
  including MODEL-CHECKER .
  including LTL-SIMPLIFIER .
endm

```

Figure 10. Check the LTL Termination Property



```

Core Maude 2.6
-----
Welcome to Maude
-----
Maude 2.6 built: Mar 31 2011 23:36:02
Copyright 1997-2010 SRI International
Fri Apr 7 15:33:10 2023

Maude> in IOD-CHECK.maude
Reading in file: "IOD-PREDS.maude"
Reading in file: "model-checker.maude"
=====
fmod LTL
fmod LTL-SIMPLIFIER
fmod SAT-SOLVER
fmod SATISFACTION
fmod MODEL-CHECKER
Done reading in file: "model-checker.maude"
Reading in file: "onlineBookStore.maude"
=====
Reading in file: "BASIC-IOD.maude"
=====
fmod BASIC-IOD
Done reading in file: "BASIC-IOD.maude"
=====
mod OrderBooks
=====
mod ConnectIoSystem
=====
mod RegisterOrder
=====
mod OrderPayment
=====
mod OnlineBookStore
Done reading in file: "OnlineBookStore.maude"
=====
mod IOD-PREDS
Done reading in file: "IOD-PREDS.maude"
=====
mod IOD-CHECK
=====
reduce in IOD-CHECK : modelCheck(OnlineBookStore, [I<> Final(< Final : FinalNode >)] .
rewrites: 400 in 16346683478ms cpu (38ms real) (0 rewrites/second)
result Bool: true
Maude> _

```

Figure 11. LTL Termination Property Result

8. Conclusion

In this work, we have proposed a formal and an MDE-based approach to tackle UML IOD formalization and analysis using the Maude language. We have defined a subset of aspects of IODs using the EMF framework in the Eclipse environment. Further, we have used the Sirius framework to develop a visual modeling tool for editing and manipulating IOD models. Acceleo is a template-based framework for model transformation and code generation technology. Specifying the IOD early in the development cycle can help identify issues and gaps in system requirements. The choice of the RL and Maude language made the analysis techniques and verification tools accessible. We have shown how using Maude allows for simulation and execution analysis. We have also exploited the underlying LTL model checker to verify the diagram's correctness. In our case, we have checked the termination property using LTL formulas. The performance of the proposed approach has been evaluated through a virtual bookstore example. In this paper, we have mainly addressed the formalization of SD-type interaction nodes. In future works, we plan to embrace the proposed work for complex concurrent control flow paths and different types of interaction diagrams in IOD interaction nodes. We also plan to check other properties using Maude model checking and annotate the analysis results in the UML IOD diagram.

9. Tool and Acceleo templates

The program code and Acceleo templates in this work are publicly available through the GitHub repository¹, necessary to run and execute for interpreting, replicating, and building on the findings reported in the paper.

Acknowledgements

The authors thank the reviewer(s) for their insightful comments and suggestions. The authors are also grateful to the Editor-in-Chief, the Editor, and the Editorial Office Assistant(s) for managing this manuscript.

References

- [1] Andrade E., Maciel P., Callou G., Nogueira B.: Mapping UML Interaction Overview Diagram to Time Petri Net for Analysis and Verification of Embedded Real-Time Systems with Energy Constraints. In: *2008 International Conference on Computational Intelligence for Modelling Control & Automation*, pp. 615–620, IEEE, 2008. doi: 10.1109/cimca.2008.44.
- [2] Baresi L., Morzenti A., Motta A., Rossi M.: From Interaction Overview Diagrams to Temporal Logic. In: *Models in Software Engineering. Workshops and Symposia at MoDELS 2010, Oslo, Norway, October 3–8, 2010, Reports and Revised Selected Papers*, pp. 90–104, Springer, 2010. doi: 10.1007/978-3-642-21210-9_9.

¹<https://github.com/IODFormalization/IOD.TO-MAUDE>

- [3] Bennama M., Bouabana-Tebibel T.: Validation environment of UML2 IOD based on hierarchical coloured Petri nets, *International Journal of Computer Applications in Technology*, vol. 47(2-3), pp. 227-240, 2013. doi: 10.1504/ijcat.2013.054372.
- [4] Bernardi S., Donatelli S., Merseguer J.: From UML Sequence Diagrams and Statecharts to Analysable Petri Net Models. In: *WOSP'02: Proceedings of the 3rd International Workshop on Software and Performance*, pp. 35-45, 2002. doi: 10.1145/584369.584376.
- [5] Bernardi S., Merseguer J.: Performance evaluation of UML design with Stochastic Well-formed Nets, *Journal of Systems and Software*, vol. 80(11), pp. 1843-1865, 2007. doi: 10.1016/j.jss.2007.02.029.
- [6] Bouabana-Tebibel T.: Semantics of the interaction overview diagram. In: *2009 IEEE International Conference on Information Reuse & Integration*, pp. 278-283, IEEE, 2009. doi: 10.1109/iri.2009.5211565.
- [7] Bowen J.P., He J.: An algebraic approach to hardware compilation, *Modern Formal Methods and Applications*, pp. 151-176, 2006.
- [8] Bruni R., Meseguer J.: Semantic foundations for generalized rewrite theories, *Theoretical Computer Science*, vol. 360(1-3), pp. 386-414, 2006. doi: 10.1016/j.tcs.2006.04.012.
- [9] Clavel M., Durán F., Eker S., Escobar S., Lincoln P., Martí-Oliet N., Meseguer J., et al.: Maude manual (version 3.1), 2020. SRI International University of Illinois at Urbana-Champaign. <http://maude.lcc.uma.es/maude31-manual-html/maude-manual.html>.
- [10] Clavel M., Durán F., Hendrix J., Lucas S., Meseguer J., Ölveczky P.: The Maude formal tool environment. In: T. Mossakowski, U. Montanari, M. Haveraaen (eds.), *Algebra and Coalgebra in Computer Science: Second International Conference, CALCO 2007, Bergen, Norway, August 20-24, 2007. Proceedings*, pp. 173-178, Springer, 2007. doi: 10.1007/978-3-540-73859-6_12.
- [11] Djaoui C., Kerkouche E., Chaoui A., Khalfaoui K.: A graph transformation approach to generate analysable maude specifications from UML interaction overview diagrams. In: *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pp. 511-517, IEEE, 2018. doi: 10.1109/iri.2018.00081.
- [12] Dobing B., Parsons J.: Dimensions of UML Diagram Use: A Survey of Practitioners, *Journal of Database Management (JDM)*, vol. 19(1), pp. 1-18, 2008. doi: 10.4018/jdm.2008010101.
- [13] Durán F., Eker S., Escobar S., Martí-Oliet N., Meseguer J., Rubio R., Talcott C.: Programming and symbolic computation in Maude, *Journal of Logical and Algebraic Methods in Programming*, vol. 110, 100497, 2020. doi: 10.1016/j.jlamp.2019.100497.
- [14] Eclipse Foundation: Acceleo, homepage, [Online]. <https://www.eclipse.org/acceleo/>. Accessed November 2023.

- [15] Eclipse Foundation: EMF, homepage Eclipse Modelling Framework (EMF), [Online]. <https://www.eclipse.dev/modeling/emf/>. Accessed November 2023.
- [16] Eclipse Foundation: Sirius, homepage, [Online]. <https://www.eclipse.org/sirius/>. Accessed November 2023.
- [17] Eker S., Meseguer J., Sridharanarayanan A.: The Maude LTL model checker, *Electronic Notes in Theoretical Computer Science*, vol. 71, pp. 162–187, 2004. doi: 10.1016/s1571-0661(05)82534-4.
- [18] Frick G., Scherrer B., Müller-Glaser K.D.: Designing the software architecture of an embedded system with UML 2.0. In: *Software Architecture Description & UML Workshop*, 2004.
- [19] Hammal Y.: A Formal Semantics of UML StateCharts by Means of Timed Petri Nets. In: F. Wang (ed.), *Formal Techniques for Networked and Distributed Systems – FORTE 2005. 25th IFIP WG 6.1 International Conference, Taipei, Taiwan, October 2–5, 2005, Proceedings*, pp. 38–52, Springer, 2005. doi: 10.1007/11562436_5.
- [20] Kerkouche E., Khalfaoui K., Chaoui A.: A rewriting logic-based semantics and analysis of UML activity diagrams: a graph transformation approach, *International Journal of Computer Aided Engineering and Technology*, vol. 12(2), pp. 237–262, 2020. doi: 10.1504/ijcaet.2020.10026291.
- [21] Kloul L., Küster-Filipe J.: From Interaction Overview Diagrams to PEPA Nets, *Online Proceedings of the 4th Workshop on Process Algebras and Timed Activities (PASTA'05)*, vol. 104, 2005.
- [22] de Lara J., Vangheluwe H., Alfonseca M.: Meta-modelling and graph grammars for multi-paradigm modelling in ATOM³, *Software & Systems Modeling*, vol. 3, pp. 194–209, 2004. doi: 10.1007/s10270-003-0047-5.
- [23] Lilius J., Paltor I.P.: vUML: a Tool for Verifying UML Models. In: *14th IEEE International Conference on Automated Software Engineering*, pp. 255–258, 1999. doi: 10.1109/ASE.1999.802301.
- [24] Louati A., Jerad C., Barkaoui K.: On CPN-based verification of hierarchical formalization of UML 2 Interaction Overview Diagrams. In: *2013 5th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO)*, pp. 1–6, IEEE, 2013. doi: 10.1109/icmsao.2013.6552703.
- [25] McUmber W.E., Cheng B.H.: A general framework for formalizing UML with formal languages. In: *Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001*, pp. 433–442, IEEE, 2001.
- [26] Meseguer J.: Rewriting Logic and Maude: A Wide-Spectrum Semantic Framework for Object-Based Distributed Systems. In: S.F. Smith, C.L. Talcott (eds.), *IFIP TC6/WG6.1. Fourth International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS 2000) September 6–8, 2000, Stanford, California, USA*, pp. 89–117, Springer, 2000. doi: 10.1007/978-0-387-35520-7_5.

- [27] Meseguer J.: Specifying, Analyzing and Programming Communication Systems in Maude. In: G. Hommel (ed.), *Communication-Based Systems: Proceeding of the 3rd International Workshop held at the TU Berlin, Germany, 31 March–1 April 2000*, pp. 93–101, Springer, 2000. doi: 10.1007/978-94-015-9608-4_7.
- [28] Meseguer J.: Twenty years of rewriting logic, *The Journal of Logic and Algebraic Programming*, vol. 81(7–8), pp. 721–781, 2012. doi: 10.1016/j.jlap.2012.06.003.
- [29] Mishra A.: Dynamic Slicing of UML Interaction Overview Diagram. In: *2019 IEEE 9th International Conference on Advanced Computing (IACC)*, pp. 125–132, IEEE, 2019. doi: 10.1109/iacc48062.2019.8971586.
- [30] Padua D. (ed.): *Encyclopedia of Parallel Computing*, Springer Science & Business Media, New York, NY, 2011. doi: 10.1007/978-0-387-09766-4.
- [31] Platt R., Thompson N.: The evolution of UML. In: *Encyclopedia of Information Science and Technology*, Third Edition, pp. 348–353, IGI Global, 2015.
- [32] Rumbaugh J., Jacobson I., Booch G.: *The Unified Modeling Language Reference Manual*, 2nd Ed., Addison Wesley Longman Ltd., 2004.
- [33] Steinberg D., Budinsky F., Paternostro M., Merks E.: *EMF: eclipse modeling framework*, Pearson Education, 2008.
- [34] Störrle H., Hausmann J.H.: Towards a formal semantics of UML 2.0 activities. In: *Software Engineering 2005*, pp. 117–128, Gesellschaft für Informatik eV, 2005.
- [35] Wazlawick R.S.: *Object-oriented analysis and design for information systems: modeling with UML, OCL, and IFML*, Elsevier, 2014.
- [36] Whittle J.: Extending interaction overview diagrams with activity diagram constructs, *Software & Systems Modeling*, vol. 9, pp. 203–224, 2010.

Affiliations

Chafika Djaoui

Mohamed Seddik Ben Yahia University, Department of Computer Science, Jijel, Algeria;
MISC Laboratory, Department of Computer Science and Its Applications, Constantine,
Algeria, c.djaoui@univ-jijel.dz

Allaoua Chaoui

University Constantine 2-Abdelhamid Mehri, MISC Laboratory, Department of Computer
Science and Its Applications, Faculty of NTIC, Constantine, Algeria,
allaoua.chaoui@univ-constantine2.dz

Received: 17.12.2023

Revised: 6.05.2024

Accepted: 22.05.2024

ALDIN KOVAČEVIĆ
MUZAFER SARAČEVIĆ
AMOR HASIĆ

BIOMETRICS-BASED GENERATION OF DIFFIE–HELLMAN KEY EXCHANGE PARAMETERS

Abstract	<i>When two parties need to securely communicate over an insecure channel, Diffie–Hellman is often employed as the key exchange algorithm. This paper presents two novel approaches to generating Diffie–Hellman parameters for key exchange based on user biometrics, namely their fingerprint data. Fingerprint templates are extracted as bit strings via a fingerprint scanner and later used as inputs. In one approach, the whole fingerprint template is utilized as a user’s private key. In the second approach, fingerprint data is scrambled into smaller chunks and rearranged into two strings that serve as the user’s private key and the basis for prime p. Both approaches were implemented and tested experimentally. After analysis, the second approach that uses scrambled fingerprint data shows better execution times and improved security and usability considerations.</i>
Keywords	biometrics, Diffie–Hellman, key exchange algorithm, fingerprint template, prime generation
Citation	Computer Science 25(3) 2024: 421–434
Copyright	© 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

Traditional cryptography is used either to securely store data in a system or to transmit data over network channels. In this technique, a cryptographic key plays the most important role in protecting information. As the key is very large (e.g., 128, 192, and 256 bits for AES) and random, there is a need to securely generate and store the key in a secret place. For any communication, the secret key is either exchanged between two users (also referred to as parties) through a key-transport protocol (using public key cryptography) or established in such a way that both communicating parties have equal influence on the key that is agreed upon. Oftentimes, the involved parties need to communicate through an insecure or public channel. Hence, the need arises for an algorithm that can efficiently generate a shared secret key between two parties, without them having to reveal any sensitive data. A very popular key-exchange algorithm is the Diffie–Hellman key exchange. Published in 1976 by Whitfield Diffie and Martin Hellman, it was one of the first practical examples of public key cryptography [10].

In Diffie–Hellman, the initial parameters chosen by both parties are raised to a selected power to produce decryption keys. The components of the keys are never directly transmitted, making the task of a would-be code breaker mathematically challenging. Moreover, the two parties do not need to have prior knowledge of each other, yet they can still work to produce the secret key together. To implement Diffie–Hellman, two end users need to mutually agree on positive whole numbers p and g , and respectively choose positive whole-number personal keys a and b (both being less than p). Afterward, modular exponentiation is performed to calculate public keys A and B that the parties use to individually calculate a shared secret key x . In this algorithm, the proper choice of parameters is important for the overall security of the resulting secret key. Namely, the value of p is recommended to be at least 2048 bits due to the latest security considerations [1]. Moreover, choosing too small values for private keys a and b can also lead to the shared secret being easily deducible. Therefore, research into the optimal values for these parameters is an ongoing topic.

In this paper, a potential approach to generating Diffie–Hellman parameters is presented through biometrics – namely, fingerprint data. Biometric data is reliable for authentication and can be integrated with traditional cryptography to make it stronger [11]. Biometric data can be used to manage a cryptographic key by making a strong link between a key and the user’s physiological characteristics. A cryptographic key may be derived from biometric data using any standard hash function or user-defined algorithms. While there has been research about the practicality of using biometric technology in cryptography, there has not been much focus on the generation of DH parameters using biometrics. The main aims of this paper are therefore: 1) find a practical and effective approach to generate DH key exchange parameters from user fingerprint data; 2) discuss the efficiency, practicality and limitations of biometrics-based DH parameter generation.

The rest of the paper is structured as follows: Chapter 2 presents the necessary background information regarding the Diffie–Hellman protocol and fingerprint anal-

ysis. Chapter 3 presents the relevant literature review on the topic of biometrics in cryptography. Chapter 4 describes two proposed approaches to using fingerprint data in *DF* parameter generation. In Chapter 5, the two approaches are analyzed and compared, with additional comments on their applicability and limitations. Lastly, the final chapter summarizes the main points of the paper and proposes future work.

2. Background information

Diffie–Hellman key exchange is a method of digital encryption that securely exchanges cryptographic keys between two parties over a public channel without their conversation being transmitted over the Internet. The two parties use symmetric cryptography to encrypt and decrypt their messages. To implement Diffie–Hellman, two end users, e.g. Alice and Bob, mutually agree on positive whole numbers p and g , such that p is a prime number and g is a generator of p . The generator g is a number that, when raised to positive whole-number powers less than p , never produces the same result for any two such whole numbers. The value of p may be large, but the value of g is usually small.

Once Alice and Bob have agreed on p and g in private, they choose positive whole-number personal keys a and b , which are less than the prime number modulus p . Neither user divulges their key to anyone. Next, Alice and Bob compute public keys A and B based on their keys according to Equations (1) and (2):

$$A = g^a \bmod p \quad (1)$$

$$B = g^b \bmod p \quad (2)$$

The two users can share their public keys A and B over an insecure channel. From these public keys, a number x can be generated by either user based on their keys. Alice and Bob can compute x using the following equations, respectively:

$$x = B^a \bmod p \quad (3)$$

$$x = A^b \bmod p \quad (4)$$

The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b , which are critical in the calculation of x , have not been transmitted over a public medium. Because it is a large and random number, a potential hacker has almost no chance of correctly guessing x . The two users can now choose whichever symmetric encryption algorithm, and encrypt their communications using the shared key x . A biometric is defined as a unique, measurable, biological characteristic or trait for automatically recognizing or verifying the identity of an individual [16]. Today, the most common use case for biometrics is the analysis of human characteristics for security purposes. Biometric methodology for authentication is appealing because of its handiness and possibility to offer security with non-denial. The most common biometrics used in security are fingerprint, hand,

iris, face and voice. Research has shown that fingerprints are suitable as long-term markers of human identity. They are detailed, unique, difficult to alter, and durable over the life of an individual, making them ideal for authentication purposes. The analysis of fingerprints for matching purposes generally requires the comparison of several features of the print pattern. These include patterns, which are aggregate characteristics of ridges, and minutiae points, which are unique features found within the patterns. The three basic patterns of fingerprint ridges that constitute the majority of all fingerprints are the loop, whorl and arch, shown in Figure 1. Other common fingerprint patterns include the tented arch, the plain arch, and the central pocket loop.



Figure 1. Most common fingerprint patterns [16]

Minutiae points are the major features of a fingerprint image and are used in the matching of fingerprints. These minutiae points are used to determine the uniqueness of a fingerprint image. A good quality fingerprint image may contain around 25 to 80 minutiae depending on the fingerprint scanner resolution and the placement of the finger on the sensor [17]. The major minutiae features of fingerprint ridges are ridge ending, bifurcation, and short ridge (or dot) as shown in Figure 2.

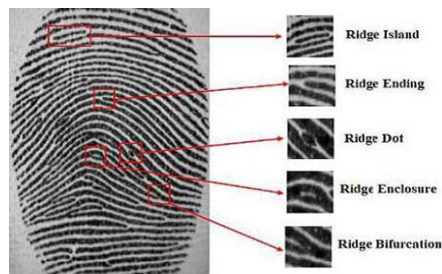


Figure 2. Most common minutiae points [16]

The ridge ending is the point at which a ridge terminates. Bifurcations are points at which a single ridge splits into two ridges. Short ridges (or dots) are ridges that are significantly shorter than the average ridge length on the fingerprint. Minutiae and patterns are very important in the analysis of fingerprints since no two fingers are

identical so far. To acquire a fingerprint as an image, a scanner system is required. In general, the fingerprint image is not saved; instead, it is converted into binary code which is used for verification. This binary code is created from the minutiae that are extracted from the fingerprint and known as the fingerprint template (see Fig. 3). Depending on the scanner manufacturer and the algorithms used, these templates can vary across devices. The algorithm cannot be used to re-convert the binary data to an image, so no one can duplicate your fingerprints.

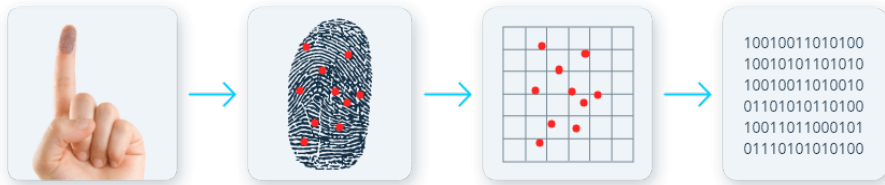


Figure 3. The process of extracting binary data from a fingerprint [9]

3. Literature review

Biometrics in cryptography has been an ongoing research topic, with many works being done in the field. In his paper, Sakre [16] introduces a novel method for exchanging symmetric keys based on personal fingerprint payloads using the SHA-1 hash algorithm. The key is then securely delivered to the other parties via an asymmetric cryptosystem. Barman et al. [5] introduce a key-exchange protocol using the biometric data of the sender and receiver. Users register their biometric information on a central server, which facilitates contact between registered users. Using a biometrics-based cryptographic construction, a user produces a cryptographic key at random and distributes it to another user. Both the sender and the receiver are guaranteed the confidentiality of the biometric information.

Wang et al. [20] propose a Diffie–Hellman key exchange and secret sharing-based fingerprint authentication method that protects user privacy. In order to securely distribute fragments of important private information among a distributed network or group, they use a secret sharing scheme, which lessens the workload on the template storage center (TSC) and the users. The user’s original fingerprint template is kept in ciphertext format in TSC to ensure the security of template data. To further safeguard the privacy of the user’s data, the DH key exchange protocol enables TSC and the user to encrypt the fingerprint template in each query using a unique one-time random key.

Juels and Sudan [12] proposed a cryptographic construction called fuzzy vault construct. The authors presented its application for a fingerprint-based security system, called fingerprint fuzzy vault. The fundamental idea is to conceal the cryptographic key in a list that has been jumbled and is made up of real fingerprint traits and made-up chaff features. The fuzzy vault’s security is based on the difficulty of the polynomial reconstruction problem.

Ueshige and Sakurai [18] proposed a one-time authentication protocol that can create biometric authentication-based secure sessions. Both the fresh biometric data and the saved templates are subjected to a one-time transformation that is specific to the session. To verify the subject's authenticity, a comparison of the two altered templates is made. Bringer et al. [7] employed the Goldwasser–Micali cryptosystem for biometric authentication. With the help of this system, the biometric comparison can be done in an encrypted domain. The system ensures that the biometric data saved in the database cannot be explicitly linked to any user identity; instead, it just checks to see if the data associated with an identity is present. Barni et al. [6] proposed a scheme for privacy-preserving authentication based on fingerprints. The ElGamal cryptosystem, which enables biometric comparison in encrypted domains, is used in this technique. Upmanyu et al. [19] proposed a blind authentication protocol that is based on homomorphic encryption. The drawback of these authentication protocols is that they can only authenticate the subject, but they cannot produce the cryptographic keys required for secure communication.

The “Secure Ad-hoc Pairing with Biometrics: SAFe” protocol proposed by Buhan et al. [8] uses the fuzzy extractor scheme and can be used to establish a secure link between two parties. Unlike many biometrics-based protocols, this protocol does not involve a biometric template database or server. However, its drawback is that it shares the biometric data between the two parties and requires mutual trust among them. Additionally, a secure channel is needed for the exchange of biometric information.

4. Proposed methodology

The following chapter explains how fingerprint data is extracted, and proposes two different approaches to utilizing said data in DH parameter generation. Before any key generation is attempted, it is necessary to extract the fingerprint data in a usable format. This process, and the size of the fingerprint template, can vary depending on the fingerprint scanner model and manufacturer. In this research, a very common fingerprint module was used – FPM10A (Fig. 4).



Figure 4. FPM10A fingerprint scanner

The module was connected to a simple Arduino setup to facilitate accessing the fingerprint templates in later steps.

This module (as well as many other scanners available on the market) is compatible with the Adafruit fingerprint scanner library. In the Adafruit library, fingerprints are converted into a 512 byte model, which is composed of 2×256 byte fingerprint templates. Upon finger enrollment, a user is asked to scan their fingerprint twice, which creates two templates that make up the model. In total, this is 4096 bits. This binary length is satisfactory for further parameter generation, as the recommended size of parameter p is at least 2048 bits [1]. Before continuing with the explanation of the proposed approach, it is worth noting that all features of the fingerprint binary string are handled by the Adafruit library, and may be different in other scanners (but the main ideas remain the same).

In terms of generable parameters, the values of the prime number p , generator g , and private keys a and b are chosen by the participating parties (p and g in agreement, a and b privately). Public keys A and B , as well as the shared secret x , are all calculated based on the initial parameters.

For the research, two parties who wish to communicate – Alice and Bob – will be introduced. As there are no special constraints on private keys, except sufficient length, their private keys a and b will be calculated from their respective fingerprint templates. While a different generator g can be chosen, the best practice recommendation is to keep g at a low value for simplicity [13]. Therefore, g will be equal to 2. However, generating the prime p will not be as straightforward. According to best practices, it should be a prime number that is greater than the private keys of both parties.

Generally, parameter p is agreed up prior to the key exchange, and any generated private keys will be bound by it. But, since the parties do not know the binary values of their fingerprint templates in advance, a special approach should be employed to make sure a large enough p is chosen of the remainder of the Diffie–Hellman algorithm. The paper proposes two different solutions.

Approach 1. Generating a and b from biometrics, and keeping p as a random prime: In the first approach, fingerprint binary strings from Alice (a) and Bob (b) are used in full as their private keys, resulting in 4096-bit keys. The prime p will be a randomly generated prime number that is larger than both a and b . Since Alice and Bob obviously cannot disclose their private keys to agree on the value of prime p , they will each propose a candidate prime p that is greater than their private key. After exchanging the candidate primes, the larger prime will be chosen as p , as it is guaranteed to be larger than both private keys. Once p is selected, the Diffie–Hellman algorithm can proceed per usual rules. Step by step, the algorithms would work as follows:

1. Alice and Bob use their fingerprint templates as their private keys a and b .
2. Alice and Bob each generate a random *candidate prime* p_a/p_b in the following way: A bit string starting with 1 and ending with 1 is generated, of length 4096. This is done to maximize the likelihood of generating a prime number since even number strings end with 0.

3. If the bit string is a number greater than the private key, proceed to the next step. If it is not, keep regenerating the bit string.
4. After the bit string is obtained, check if it is a *probable prime*, using Miller–Rabin [15] and Lucas–Lehmer [14] tests.
5. If it is a prime, proceed to the next step. If it is not, find the next probable prime higher than that number (using the same primary tests from the previous step).
6. After Alice and Bob generate their candidate primes p_a and p_b they exchange them, and the larger of the two becomes the prime p that will be used.
7. After the generation of p , all necessary parameters are obtained, and Diffie–Hellman can continue as usual. Figure 5 showcases a sample key exchange using this approach.

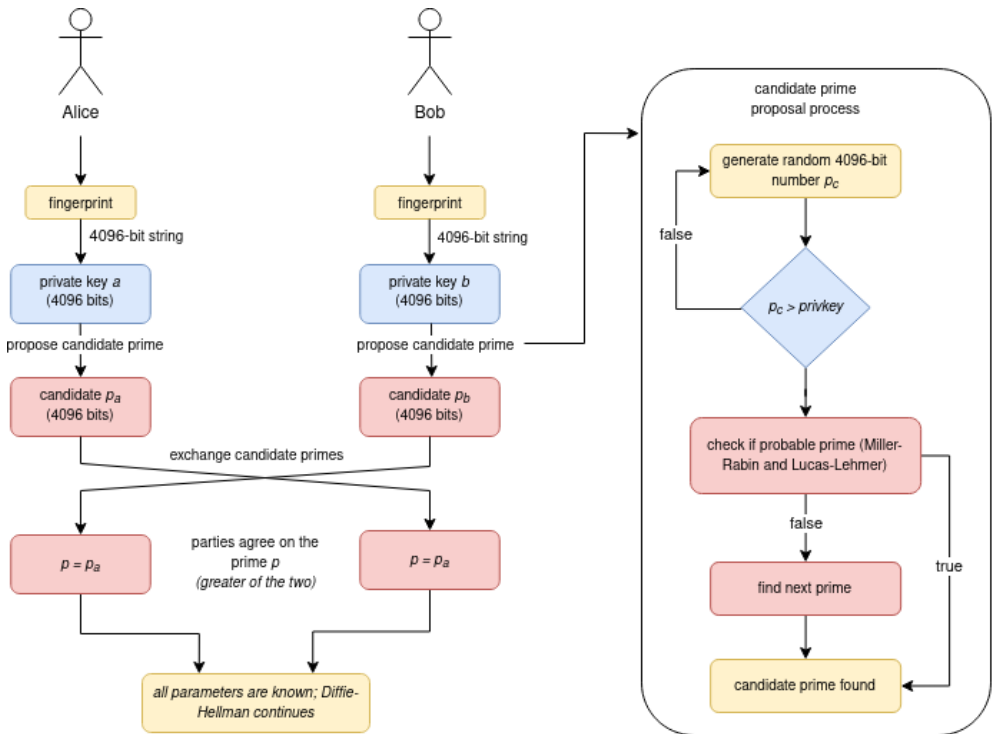


Figure 5. Approach 1. Generating a and b from biometrics

The benefits of this approach are that the entire fingerprint template is used as the user's private key, and the resulting secret key is very large (4096 bits). On the other hand, prime generation at this scale is a computationally challenging task, and finding p may prove to be a slow process. Moreover, in this approach, biometrics are only used for the private keys. With these considerations in mind, the authors propose another solution that employs biometrics in both the private keys, and generation of prime p .

Approach 2. Generating a, b and p from biometrics. While using all 4096 bits of a fingerprint template does result in stronger keys, research has shown that 2048-bit keys offer sufficient practical security. Therefore, if only 2048 bits of a fingerprint were used in the private key, the other half could be utilized in the generation of prime p . Naturally since p is a parameter that needs to be exchanged between the parties, it would not be acceptable to transmit raw and private fingerprint data. Instead, in this approach, the fingerprint would be randomly scrambled into chunks of 32 bits each, resulting in 128 chunks. Also, 64 random chunks (2048 bits) are then used as the private key (a or b). The other 2048 bits are used as the basis of the user's candidate prime p_c – either as is (if the number is prime) or the next prime number is picked. The rest of the algorithm would function the same as Approach 1: Alice and Bob would exchange their candidate primes, the larger of them would be selected as p , and Diffie–Hellman could proceed. A diagram of this flow can be found in Figure 6.

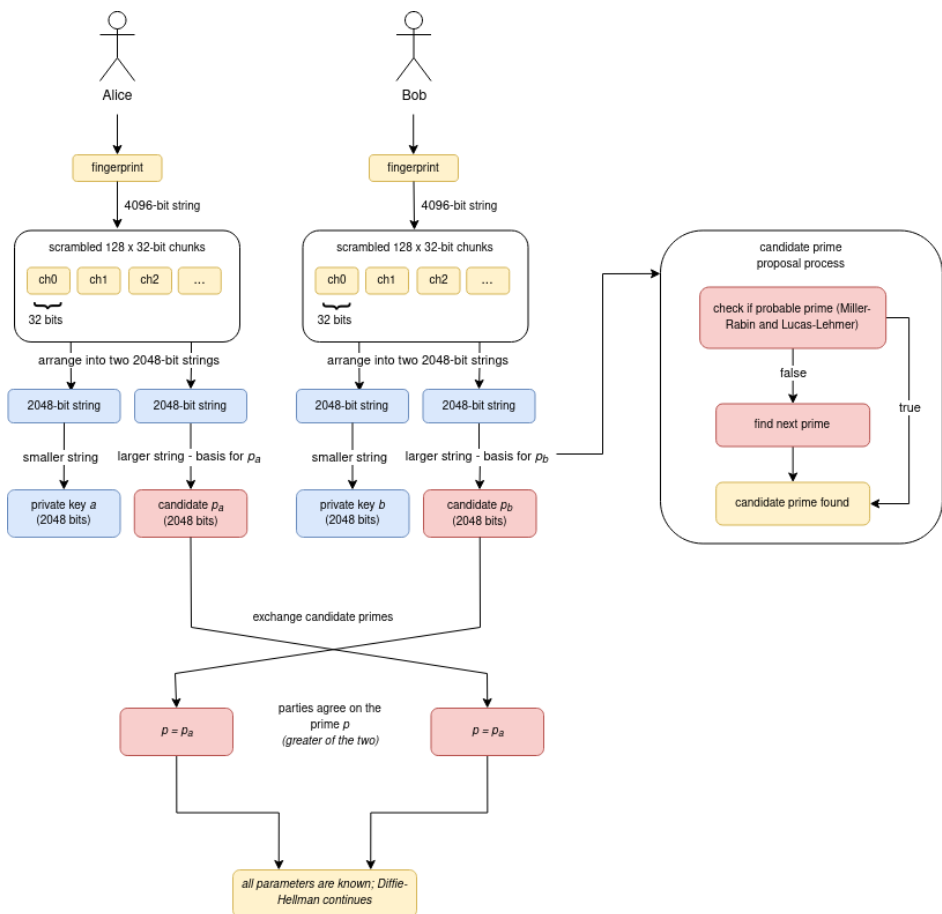


Figure 6. Approach 2. Generating a, b and p from biometrics

The following paragraphs outlines the steps involved in this approach:

1. The fingerprint templates of Alice and Bob (respectively) are scrambled into 128 32-bit chunks.
2. Two-bit strings of 2048 bits each are constructed at both ends.
3. The lesser 2048-bit string is selected to be used as the private key a / b . The other (greater) 2048-bit string will be used as the basis of candidate prime p_a / p_b .
4. Check if the proposed bit string is a probable prime, using Miller–Rabin and Lucas–Lehmer tests.
5. If it is a prime, proceed to the next step. If it is not, find the next probable prime from that number (using the same primary tests from the previous step).
6. After Alice and Bob generate their candidate primes p_a and p_b they exchange them, and the larger of the two becomes the prime p that will be used.
7. After the generation of p , all necessary parameters are obtained, and Diffie–Hellman can continue as usual.

This approach works with smaller, albeit still secure, key values which can result in faster generation. Moreover, it maximizes the utilization of biometrics, using fingerprint data to produce three out of four generable parameters (everything except g , which is always agreed to be a small value).

5. Results and discussion

Both approaches were experimentally implemented in Java, with the BigInteger library used for all mathematical operations. The example code has been made available on GitHub.

5.1. Experimental results

Table 1 below compares the runtimes of individual steps for each approach. The average of 10 runs is taken as the average time, and tests were executed on an AMD Ryzen 7 4800H CPU with 32 GBs of RAM.

Table 1
Time taken for different operations in both approaches

Operation	Approach 1 [ms]	Approach 2 [ms]
Private key generation	0.85	5.85
Candidate prime generation	5367.85	571.65
Selection of prime p	0.02	0.02
Public key generation	45.10	6.95
Secret key generation	39.55	7.20
Total	5453.37	591.67

As can be seen, the 2nd approach is more performant, averaging at around 9.22 times faster execution of Diffie–Hellman, compared to the 1st approach. All

individual operations in the 2nd approach are also faster, except for private key generation. This is explained by the fact that the second algorithm needs to spend some time to scramble the fingerprint template data, and choose the correct 2048-bit string as the private key.

The main bottleneck in both algorithms is the generation of the candidate prime. Generation of primes at the scale required (4096 and 2048 bits) is a computationally expensive process, which leads to increased run times. Expectedly, the bottleneck is less disruptive in the 2nd approach, as it needs to generate a much smaller prime than the 1st approach. If the communicating parties wish to generate the shared secret key once, and conduct all future correspondence using the same key, slower generation times might not be an issue. However, nowadays Diffie–Hellman is most often used in its “ephemeral” variant. In ephemeral Diffie–Hellman, the key exchange is at “session-level”, with each new session using different starting parameters and resulting in a new secret key (the old keys are discarded). In that case, where real-time performance is a requirement, the 2nd approach might be preferred.

5.2. N -bit security

In terms of key length, both approaches offer appropriate security. The term n -bit security refers to the property of an algorithm in which an attacker would have to perform an average of $2n$ operations to break it [2]. Generally, attacks that take more than 2100 operations to break an algorithm are considered to be far too impractical to conduct [3]. For symmetric keys, their size should be at least $2n$, whereas algorithms that rely on modular exponentiation / prime numbers should have much larger key sizes. According to NIST (National Institute of Standards and Technology) recommendations [4], to achieve 112-bit security (comparable to symmetric 3DES), the private key should comprise at least 224 bits, and p should be at least 2048 bits. Moreover, for 128-bit security (comparable to AES-128), key sizes should be 256 bits for the private key, and 3072 bits for p . It can be seen that both approaches satisfy 112-bit security. 224-bit security is satisfied by the 1st approach (due to using 4096-bit primes), but the 2nd approach falls slightly short due to its 2048-bit prime size. This approach could potentially be modified to use 1024 bits for the private key, and the remaining 3072 for the prime p , thereby satisfying 128-bit security at the cost of increased computation time.

5.3. Key pool size

Another consideration is the key pool available in both approaches. If the entire fingerprint is used as the user’s private key, the user has a pool of 10 available private keys. If a private key accidentally leaked or was hacked, it could not only compromise the user’s conversations but potentially their identity as well. On the other hand, the 2nd approach uses scrambled data from a fingerprint, not the original bit string. This ensures that, even if a private key were to leak, the user’s biometrics would still be protected, as it would not be possible to reconstruct the actual fingerprint template

from available data. Moreover, since the private key uses 64 chunks of 32 bits (the other 64 chunks are used in the generation of prime p), the possible pool of keys is $128!/64!$, which is approximately 3.04×10^{126} possible permutations.

All in all, while both approaches are viable, the 2nd approach appears to have more practical benefits. While the 1st approach offers slightly better n -bit security, the 2nd approach is more computationally efficient, has a larger key pool and offers more precautions against user biometrics leaking.

6. Conclusion

To summarize, the result of this research paper is the proposal of two different algorithms for the generation of Diffie–Hellman key exchange parameters, based on fingerprint biometrics. Initially, the fingerprint templates are extracted as 4096-bit strings using an Adafruit-compatible fingerprint scanner. Afterward, two approaches are described. In the first approach, fingerprint templates are used as the users' private keys, and the p is a random prime number found to be greater than both fingerprint bit strings. In the second approach, the fingerprint templates are scrambled into 128 chunks of 32 bits each. These chunks are then assembled into 2048-bit bit strings, one of which is used as the private key, and the other as the basis for the generation of prime p . The prototypes for both algorithms are developed in Java, showing them to be feasible. After computational and security analyses, the second approach was found to be more advantageous, offering increased performance and a greater key pool over the first approach.

In the future, the authors plan to improve upon the second proposed approach and come up with a way to speed up prime generation, making the algorithm more usable in practice. The authors believe this paper presents a viable case for the usage of fingerprint biometrics in Diffie–Hellman that warrants additional future consideration and experimentation.

References

- [1] Adrian D., Bhargavan K., Durumeric Z., Gaudry P., Green M., Halderman J.A., Heninger N., *et al.*: Imperfect Forward Secrecy: How Diffie–Hellman Fails in Practice. In: *CCS'15: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 5–17, ACM, New York, NY, USA, 2015. doi: 10.1145/2810103.2813707.
- [2] Alwen J.: The bit-security of cryptographic primitives, AWS Wickr. <https://wickr.com/the-bit-security-of-cryptographic-primitives-2/>. Accessed September 2023.
- [3] Aumasson J.P.: *Too Much Crypt*, Real World Crypto, 2020.
- [4] Barker E., Barker W.C.: *Recommendation for key management: Part 2 – Best Practices for Key Management Organizations*, NIST: National Institute of Standards and Technology, 2019. doi: 10.6028/nist.sp.800-57pt2r1.

- [5] Barman S., Chattopadhyay S., Samanta D., Panchal G.: A novel secure key-exchange protocol using biometrics of the sender and receiver, *Computers and Electrical Engineering*, vol. 64, pp. 65–82, 2017. doi: 10.1016/j.compeleceng.2016.11.017.
- [6] Barni M., Bianchi T., Catalano D., Di Raimondo M., Donida Labati R., Failla P., Fiore D., *et al.*: Privacy-preserving fingercode authentication. In: *MM&Sec'10: Proceedings of the 12th ACM workshop on Multimedia and security*, pp. 231–240, ACM, New York, NY, USA, 2010. doi: 10.1145/1854229.1854270.
- [7] Bringer J., Chabanne H., Izabachène M., Pointcheval D., Tang Q., Zimmer S.: An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication. In: J. Pieprzyk, H. Ghodosi, E. Dawson (eds.), *Information Security and Privacy. 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2–4, 2007, Proceedings*, vol. 7, pp. 96–106, Springer, Berlin Heidelberg, 2007. doi: 10.1007/978-3-540-73458-1_8.
- [8] Buhan I.: *Cryptographic keys from noisy data: theory and applications*, PS University Press, 2008.
- [9] Durairajan M.S., Saravanan R.: Biometrics Based Key Generation using Diffie Hellman Key Exchange for Enhanced Security Mechanism, *Recent Trends in Biotechnology and Chemical Engineering*, vol. 6(9), pp. 4359–4365, 2014.
- [10] Gillis A.S.: *Diffie-Hellman key exchange (exponential key exchange)*. <https://www.techtarget.com/searchsecurity/definition/Diffie-Hellman-key-exchange>.
- [11] Ha F., Anderson R., Daugman J.: Combining crypto with biometrics effectively, *IEEE Transactions on Computers*, vol. 55(9), pp. 1081–1088, 2006. doi: 10.1109/tc.2006.138.
- [12] Juels A., Sudan M.: A fuzzy vault scheme, *Designs, Codes and Cryptography*, vol. 38(2), pp. 237–257, 2006. doi: 10.1007/s10623-005-6343-z.
- [13] Kivinen T., Kojo M.: *RFC3526: More modular exponential (MODP) Diffie–Hellman groups for Internet Key Exchange (IKE)*, IETF Datatracker.
- [14] Lucas-Lehmer Test, Prime Wiki. https://www.rieselprime.de/wiki/Lucas-Lehmer_test. Accessed September 2023.
- [15] Primality tests. <https://crypto.stanford.edu/pbc/notes/numbertheory/millerrabin.html>. Accessed September 2023.
- [16] Sakre M.M.I.: Exchanging Biometric Keys in Secrecy, *International Journal of Scientific and Engineering Research*, vol. 6(9), pp. 1113–1120, 2015.
- [17] Soheat S., Wang T.: Fingerprint Enhancement, Minutiae Extraction and Matching Techniques, *Journal of Computer and Communications*, vol. 8(5), pp. 55–74, 2020. doi: 10.4236/jcc.2020.85003.
- [18] Ueshige Y., Sakurai K.: A Proposal of One-Time Biometric Authentication. In: H.R. Arabnia, S. Aissi (eds.), *Proceedings of the 2006 International Conference on Security & Management, SAM 2006, Las Vegas, Nevada, USA, June 26–29, 2006*, pp. 78–83, CSREA Press, 2006.

- [19] Upmanyu M., Namboodiri A.M., Srinathan K., Jawahar C.V.: Blind authentication: A secure crypto-biometric verification protocol, *IEEE Transactions on Information Forensics and Security*, vol. 5(2), pp. 255–268, 2010. doi: 10.1109/TIFS.2010.2043188.
- [20] Wang H., Luo M., Ding Y.: Privacy-preserving fingerprint authentication using D-H key exchange and secret sharing, *Security and Communication Networks*, vol. 2021(1), 5344696, 2021. doi: 10.1155/2021/5344696.

Affiliations

Aldin Kovačević

International Burch University Ilidža, Department of Information Technologies, Bosnia and Herzegovina, aldin.kovacevic@ibu.edu.ba

Muzafer Saračević

University of Novi Pazar, Department of Computer Sciences Novi Pazar, Serbia, muzafer.saracevic@uninp.edu.rs

Amor Hasić

University of Novi Pazar, Serbia, amorhasic@gmail.com

Received: 31.01.2024

Revised: 15.04.2024

Accepted: 15.04.2024

DHEEPIKA PS
UMADEVI V

A NOVEL HYBRID DEEP LEARNING APPROACH FOR 3D OBJECT DETECTION AND TRACKING IN AUTONOMOUS DRIVING

Abstract

Recently Object detection and tracking using fusion of LiDAR and RGB camera for the autonomous vehicle environment is a challenging task. The existing works initiates several object detection and tracking frameworks using Artificial Intelligence (AI) algorithms. However, they were limited with high false positives and computation time issues thus lacking the performance of autonomous driving environment. The existing issues are resolved by proposing Hybrid Deep Learning based Multi Object Detection and Tracking (HDL-MODT) using sensor fusion methods. The proposed work performs fusion of solid state LiDAR, Pseudo LiDAR, and RGB camera for improving detection and tracking quality. At first, the multi-stage preprocessing is done in which noise removal is performed using Adaptive Fuzzy Filter (A-Fuzzy). The pre-processed fused image is then provided for instance segmentation to reduce the classification and tracking complexity. For that, the proposed work adopts Lightweight General Adversarial Networks (LGAN). The segmented image is provided for object detection and tracking using HDL. For reducing the complexity, the proposed work utilized VGG-16 for feature extraction which forms the feature vectors. The features vectors are then provided for object detection using YOLOv4. Finally, the detected objects were tracked using Improved Unscented Kalman Filter (IUKF) and mapping the vehicles using time based mapping by considering their RFID, velocity, location, dimension and unique ID. The simulation of the proposed work is carried out using MATLAB R2020a simulation tool and performance of the proposed work is compared with several metrics that show that the proposed work outperforms than the existing works.

Keywords

3D object detection, object tracking, hybrid deep learning, pre-processing, segmentation, sensor image fusion

Citation

Computer Science 25(3) 2024: 435–467

Copyright

© 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

In recent days, autonomous vehicles have developed rapidly, in which three-dimensional (3D) multiobject detection and tracking are performed [10]. This provides vital information by estimating traffic scale over time, and orientation. Various types of sensors are used to detect objects and perform tracking such as optical RADAR, cameras, and LiDAR in autonomous vehicles especially, Light Detection and Ranging (LiDAR) are equipped in autonomous vehicles to recognize and detect multiple 3D objects which provide deep measurements [2, 21]. RGB images are also captured from cameras with high resolution to detect the objects [27, 28]. Pseudo-LiDAR is used in several approaches to extract depth from the images and transform the image into 3D cloud points [19]. However, several challenges occurred in Pseudo LiDAR due to high sparse points and complexity [8, 23]. To overcome this challenge, RGB-D images and LiDAR cloud points are combined to perform multi-object detection by considering obstacles, static and dynamic objects. However, environmental and other factors increase the noise which reduces the detection accuracy [17, 22]. Bounding box estimation is performed by considering orientation and location of all objects present in the frame to detect the objects. In some works, image features are considered to estimate bounding boxes. However, the appearance and geometrical factors of the image are ignored which reduces the accurate object detection when it is in dynamic state (i.e., motorcycles, pedestrians, etc.) [18]. Segmentation is performed to improve classification accuracy. Various types of features are extracted from the point cloud images and RGB images which are under the category of spatial and temporal related features [3]. In several works, ground extraction is performed initially then classify the non-ground points based on several classes which reduce the complexity of object segmentation and provide high accuracy. However, it reduces the speed which is not applicable for real-time scenarios. Various deep learning neural networks are used to detect and track objects such as convolutional neural networks (CNN), residual neural network (ResNet), you only look once (YOLO), single-shot detector (SSD), etc., in which YOLO and SSD are single-stage classification algorithms whereas CNN, ResNet algorithms are two-stage algorithms [9]. Initially, these algorithms are used to detect 2D objects. Later the 2D cloud points are converted into 3D voxels using 3D CNN for detecting 3D objects [13]. The point cloud images are projected in bird's eye view in some works to know the dense of the image. However, it consists of several challenges such as occlusion of objects and perspective-related problems. However, single-stage algorithms do not detect small objects accurately whereas two-stage algorithms do not applicable for real-time scenarios due to low speed [31].

Acronyms

LIDAR	–	Light Detection and Ranging
RGB-D	–	Red Green Blue-Depth
HDL-MODT	–	Hybrid Deep Learning based Multi Object Detection and Tracking

A FUZZY	– Adaptive Fuzzy Filter
MSO	– Moth Swarm Optimization
LGAN	– Lightweight General Adversarial Networks
VGG	– Visual Geometry Group
HDL	– Hybrid Deep Learning
YOLO	– You Only Look Once
IUKF	– Improved Unscented Kalman Filter
RFID	– Radio-Frequency IDentification
RADAR	– Radio Detection And Ranging
CNN	– Convolutional Neural Networks
ResNet	– Residual Neural Network
SSD	– Single-Shot Detector
KITTI	– Karlsruhe Institute of Technology and Toyota Technological Institute

1.1. Motivation and objectives

The 3D object detection and tracking method faced many shortcomings in terms of Less detection, classification accuracy, and less tracking accuracy. The existing works provided some approaches however, they failed to provide precise results. Some of the shortcomings faced by the existing works are:

- **Less Detection Accuracy.** The existing works directly acquire images from the datasets and undergo further processes which limit them with less detection accuracy as the directly acquired images suffer from noise and poor quality. Also, the existing works employ single stage detector for object detection, which also limits the detection accuracy.
- **Complexity in Object Classification.** High complexity in classification affects classification accuracy. The existing works are limited with high complexity during classification as they extract raw features directly from the images without performing segmentation.
- **Poor Object Tracking.** The existing works lack poor tracking accuracy as they consider only current and previous time stamps for object tracking however, some of the object tracking-related metrics are not considered.

The above major pitfalls motivated us to deliver a robust, reliable, and accurate framework with the aim of detecting and tracking the objects with high detection accuracy and rapid classification using Solid-state LiDAR, Pseudo LiDAR, and RGB-D images. In addition, various problems are addressed in this research based on sparsity of cloud points, false positive rates, etc.

The foremost objective of this work is to detect and track the 3D objects using solid-state LiDAR, pseudo-LiDAR, and RGB-D images with high detection accuracy and rapid classification for efficient tracking. The remaining objectives of this proposed work are sorted below.

- To enhance the quality of LiDAR and RGB-D images, pre-processing is performed for removing the noise and equalizing the luminance effect which increases the accuracy of the detection results.
- To improve the viewport prediction of the input image by performing image rotation and instance segmentation for detecting the objects' pose accurately even for small objects which improves the tracking reliability.
- To reduce the false-positive rate, extraction of multiple features is performed which extracts numerous features to increase the accuracy of detection and tracking.

1.2. Research contribution

Designing a highly accurate 3D object detection and tracking model for autonomous driving using deep learning algorithm is the major aim of this work. Some of the research contributions are provided below:

- The problem of image quality, noise factors, and less contrastness are resolved by performing multi stage pre-processing. The existing works performs only noise removal as pre-processing technique. The proposed work performs multi-stage pre-processing as A-Fuzzy based noise removal, MSO based contrast enhancement, and point to voxel conversion.
- The complexity issues during object detection and tracking are resolved by performing instance segmentation using LGAN algorithm. Most of the existing works provides the raw data to the classifier which increases the computation of object detection and classification respectively.
- The accuracy of the object detection and tracking is improved by adopting HDL algorithm in which VGG 16 is utilized for feature extraction, and YOLOv4 is utilized for object detection and classification. Further, the object tracking is achieved by adopting IUKF algorithm based on RFID, unique ID, location, velocity, and dimension.

1.3. Paper organization

The rest of this paper is organized as follows; the section II provides the literature survey along with the existing gaps. Section III emphasizes the problem statement which shows the major research works and their corresponding problems. Section IV details the proposed work with detailed explanation along with diagrams and pseudocodes. Section V explains the experimental analysis in which four sub-sections provides such as simulation setup, dataset description, experimental analysis, and research summary. Section VI concludes the proposed work.

2. Literature survey

This section emphasizes the existing literatures and gaps associated with them in object detection and tracking for autonomous driving. Furthermore, this section also subdivided into three sections which are also listed below.

2.1. Object detection approaches

Authors in this work introduce camera image-based 3D object detection [29]. This work extracts Pseudo LiDAR points from stereo images and performs object classification which give low cost for object detection however, the Pseudo LiDAR points are prone to high sparsity.

Authors in this work perform segmentation of foreground-based object detection from the LiDAR cloud points [24]. Here raw LiDAR point clouds were taken as input for real-time object detection however, the LiDAR sensor is prone to noise and environmental conditions which leads to less detection accuracy.

Authors in this work perform autonomous vehicle detection by employing LiDAR point clouds [5]. This work extracts feature from the raw LiDAR point clouds however, the extraction of features from the raw LiDAR point clouds leads to high complexity in object classification.

The 3D object detection for autonomous vehicles was performed using fusion of LiDAR and camera data approach was discussed in [32]. This work utilized convolutional neural network for 3D object detection however, the convolutional neural network is limited with feature redundancy which leads to high complexity during object classification.

2.2. Object tracking approaches

Authors in this paper introduced 3D probabilistic object tracking model for autonomous driving [4]. This work considers both camera and LiDAR images for 3D object tracking. Based on the matching result, the object tracking was performed and unmatched results were further provided to initialization of tracking phase.

Authors in this work perform a 3D object tracking framework by introducing SIMTRACK [15]. This work performs both object detection and classification by considering only LiDAR images as input. This work attains less detection accuracy and poor tracking as they considered only raw LiDAR point clouds for object detection, and considered only current and previous time stamps for object tracking respectively.

In this paper [25], author proposed an approach to perform tracking of multiple targets for autonomous vehicle environment using YOLOv3. Experimental analysis is performed using two datasets namely KITTI and UA-DETRAC datasets in terms of processing speed and accuracy. Here, YOLOv3 based object detection and tracking was performed. However, it cannot able to detect small objects in an efficient manner which reduces the tracking efficiency.

Authors in this work [16], perform camera fusion methods for tracking objects using 3D in space. This work fused the imaging modalities such as radar and 3D camera. This work directly provides the fused data to the center fusion network without pre-processing it, that leads to lesser tracking accuracy.

2.3. Object detection and tracking approaches

Authors in this paper [20], proposed object detection and tracking for moving objects using 360-degree view camera. Here, moving object is detected based on the position and velocity, however direction is also an important metric for object tracking, hence this research obtains less performance in moving object tracking that reduces tracking accuracy.

Authors in this work [7], utilized Kalman filter method for performing object detection and tracking for autonomous vehicles. The object detection and tracking model was highly suitable for pedestrians, bicycles, and cars. The results shows that the fusion of LiDAR and radar gains better results than the radar and LiDAR only modalities.

In [1], authors perform fusion methodology for enabling object detection and tracking for autonomous vehicles. The detected objects were tracked using radar which used gaussian mixture probability hypothesis density filtering algorithm based on three phases such as booting, prediction, and update. The gaussian mixture probability density hypothesis filtering was highly linear that did not suit for real time environment.

As same as the aforementioned papers, authors in [14] also performed fusion of camera and radar for joint object detection and tracking. This work utilized faster regional convolutional neural network for object detection whereas the radar information was utilized for object tracking. Here, the utilization of faster regional convolutional neural network limits with higher time consumption and less convergence.

3. Problem statement

The major problems associated with the specific prior works are provided in this section. Furthermore, this section also provides the brief research solutions for the mentioned problems.

An accurate and effective 3D object detection framework for autonomous vehicles was introduced in this work [30]. This work consists of three phases namely fusion phase, voxel-wise feature encoder phase, and 3D backbone network phase. This work performs 3D object detection by aggregating the RGB image and LiDAR point cloud image. The LiDAR point cloud image images were voxelized in order to extract the point-wise features, and the RGB image features are extracted directly from the RGB images. Both the extracted features are aggregated in the fusion phase. Finally, the extracted voxel features are fed to 3D backbone network which consists of 3D sparse convolutional layers and performs bounding box classification.

Authors in this work introduces an object detection framework in real-time environment using LiDAR [6]. This work consists of phases such as input data acquisition, segmentation, and classification. Initially, the LiDAR point cloud data were acquired from the data set from which LiDAR point cloud map was formed. The LiDAR point cloud map was provided for segmentation in which three sub-phases are involved namely hierarchical segmentation, hierarchical merge, and extraction of ground. Finally, the Yolov4 classifies and detects the objects which were represented in 3D bounding boxes.

The major limitations associated with those works are listed below:

- This work employs LiDAR sensors for acquiring LiDAR point cloud images which were effective and accurate however, the LiDAR sensors are limited with high cost and prone to environmental conditions.
- Here, feature extraction was done during classification phase in which only limited features are extracted however, this attains poor classification as they considered only limited features (i.e., only textual features).
- The adoption of single-stage detector (i.e., Yolov4) in [6] was used for feature extraction and object detection which also limits with less detection accuracy as it did not withstand with heavy features.

A joint object detection and object tracking framework using LiDAR point clouds was introduced in this work was discussed in [26]. This work consists of four phases namely feature extraction phase, association phase, refinement phase, and trajectory phase. Initially, the two LiDAR point cloud points are taken inputs, that were provided to the feature extraction phase in which point-wise features are extracted for both the images and 3D bounding boxes were assigned. The output of the feature extraction phase was provided to association phase in which feature fusion and foreground removal were taken place. The refinement phase was used to refine the aggregated features and also provides the tracking displacement information of both frames. Finally, the trajectory phase matches the displacement of both frames and tracks the image in bird's eye view.

Authors in this work [12] utilized LiDAR and camera, joint object tracking and classification were introduced in this work. This work consists of two stages namely detection stage, and classification stage. Initially, the camera and LiDAR point cloud images with current and previous time stamps are provided to combine networks in which both the temporal and spatial information are combined. Based on the combining result, heat map was formed. The fusion network fuses both the information provided for object detection in which regional proposals and refinement of the proposals were made and performs object detection. Based on the detection and time stamps, spatiotemporal graphs were constructed. Finally, based on the graphs, the object tracking was performed in adjacent network.

The major problems centred in this work are listed below:

- Here, the raw input LiDAR input images were taken for feature extraction and object detection however, the acquisition of raw LiDAR was prone to noise, environmental conditions, and also achieves increased complexity.
- The object tracking was performed based on the current and previous timestamps by using graph neural networks however, the tracking accuracy was affected by not considering some of the tracking-related attributes (location, dimension, orientation, etc.).
- The 3D object detection was performed based on the regional proposals and refinement for the fused features however, the features fed to the classifier was not effective as it holds unnecessary background information which increases the complexity.

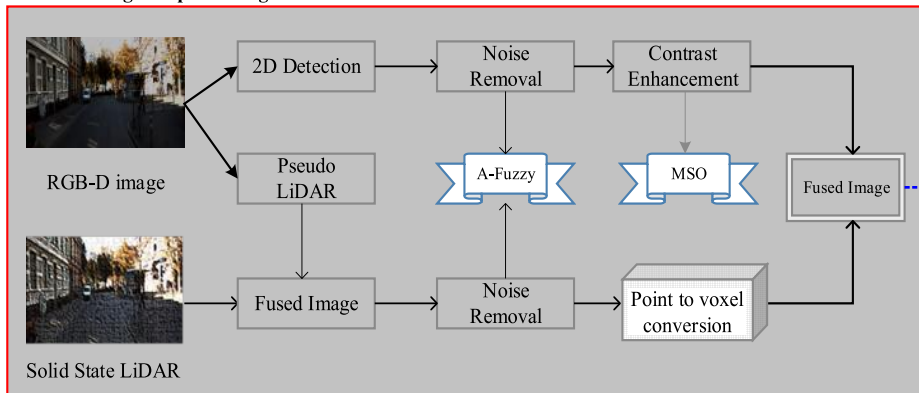
3.1. Research solutions

The aforementioned research problems are resolved by proposing 3D multi-objective object detection and tracking method using deep learning algorithm. The proposed work fused the three input images such as RGB-D, pseudo-LiDAR cloud points, and solid-state LiDAR for improving the accuracy of object detection and tracking. At first, the images acquired from the dataset are preprocessed in which the proposed work performs multi stage pre-processing which includes noise removal using A-Fuzzy, contrast enhancement using MSO, and point to voxel conversion. The pre-processed fused images are fed for segmentation to reduce the classifier complexity. As the fused images are of different orientations, the proposed work tends to manage them by rotating the images into four degrees such as 10° , 90° , 180° , and 270° . Once, all the images are properly oriented the instance segmentation is performed by L-GAN algorithm. From the segmented part, the feature extraction and classification is performed using hybrid deep learning algorithm named VGG 16 and YOLOv4 respectively. The VGG 16 is utilized for feature extraction whereas the YOLOv4 for enabling high speed and precise classification of moving object. The detected objects are then tracked based on several metrics using IUKF. Furthermore, the reliability of tracking is improved by performing mapping in location and time respectively.

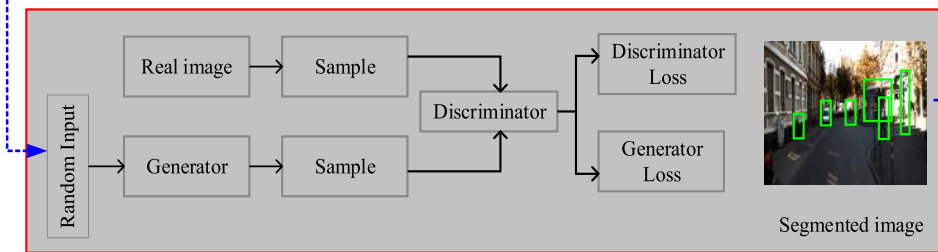
4. Proposed work

Accurate 3D object detection and tracking are mainly focused in this research for autonomous vehicle environments. For this purpose, we take input images from Solid-state LiDAR, Pseudo LiDAR, and RGB-D images. The adoption of intelligence algorithms in this work is to ensure the precision, reliability, and timeliness of the proposed framework. The proposed work adopts Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) dataset for effectively train the proposed deep learning algorithm for autonomous driving environment. The system model of the proposed work is shown in Figure 1. This proposed work consists of three sequential processes which are described as follows.

1. Multi-Stage Preprocessing



2. L-GAN based Instance Segmentation



3. 3D object Detection and Tracking

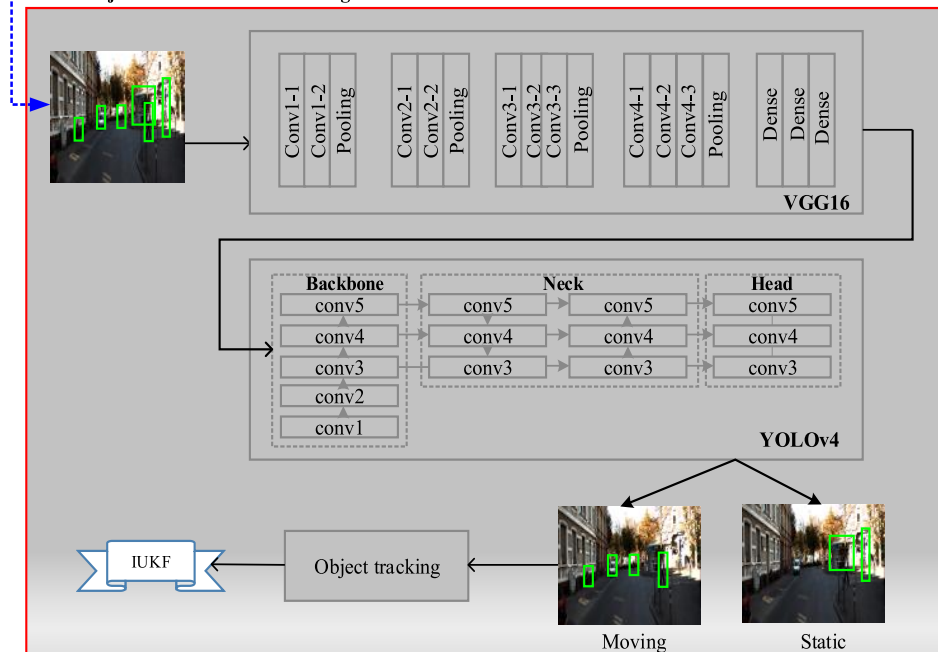


Figure 1. Overall architecture of proposed 3D object & tracking model

4.1. Multi stage pre-processing

Initially, we have taken three types of inputs, they are RGB-D images, Pseudo LiDAR cloud points, and Solid-state LiDAR cloud points in which the pseudo LiDAR cloud points are obtained from the RGB-D images. Solid-state LiDAR and Pseudo LiDAR are fused to decrease the sparsity. The Solid-state LiDAR is the emerging technology in 3D object detection that overcomes the LiDAR in terms of low cost, high speed, and better accuracy. The fusion of three inputs, cancels the disadvantages of one another thereby achieving better representation of real time scenes. The Table 1 represents the comparison of upsides and downsides of RGB-D images, Pseudo LiDAR cloud points, and Solid-state LiDAR cloud points. In addition, it illuminates the scenario with high speed.

Table 1
Comparison of proposed inputs

Proposed inputs	Upsides	Downsides
Solid-state LiDAR	<ul style="list-style-type: none"> – Compact size and structure – Free from calibration and effectively capture important features – Less cost than conventional LiDAR 	<ul style="list-style-type: none"> – Not with stand with bad climatic situations – Not suitable for omnidirectional scanning
Pseudo-LiDAR	<ul style="list-style-type: none"> – Clearer depth information – Low power when compared to LiDAR 	<ul style="list-style-type: none"> – High depth estimation error – Ineffective in capturing local features
RGB-D	<ul style="list-style-type: none"> – Provides orientation and poses of the objects – Provides omnidirectional view of the environment 	<ul style="list-style-type: none"> – Problem of interference when two cameras capture same scene – Capturing and measuring range is limited

Multi-stage preprocessing is classified into three stages which are explained as follows.

4.1.1. Noise removal

Generally, RGB-D images and LiDAR cloud points have more noises which reduce the quality of the images. To overcome this issue, noise removal is performed using A-Fuzzy with two stages. In the initial stage, the pixels are classified as noisy and good. The noisy pixels are taken in the second stage to remove the noise. In addition, this filter preserves the edge which increases the detection accuracy efficiently.

Stage 1: For pixel (pix_{ji}) of an input image, the pixel set of neighborhoods (nei_{ji}^W) is assessed which is associated with the $pix_{ji} \in K$ in which K is the input image

among that W is half filter window. Therefore, nei_{ji}^W can be formulated as,

$$nei_{ji}^W = \{pix_{j+n,i+r} \quad \forall n, r \in [-W, W]\} \quad (1)$$

From the above equation, the size of nei_{ji}^W is $(2W + 1) \cdot (2W + 1)$. For instance, if k_m is the pixel element in nei_{ji}^W then $m = 1, 2, \dots, M$ in which $M = (2W + 1) \cdot (2W + 1)$. Once, the nei_{ji}^W is computed, the membership function for every nei_{ji}^W element is computed. In this stage, the noise intensity value is determined with range of 0 to 1 i.e. $pix_{ji} \notin \{0, 1\}$. For every element k_m of nei_{ji}^W the membership function mem_{ji}^W is determined based on gaussian membership functions $\delta_{(mem_{ji}^W)}(k_m) : nei_{ji}^W \rightarrow [0, 1]$.

For the type-3 fuzzy set, the $\widetilde{(mem)}_{ji}^W$ is defined by the $\delta_{\widetilde{(mem)}_{ji}^W}(k_m, \delta_{mem_{ji}^W})$. The association of mem_{ji}^W with the gaussian membership function can be formulated as,

$$\delta_{(mem)_{ji}^{(W,n)}}(k_m) = e^{-(k_m - \vartheta_{ji}^{(W,n)})^2 / 2(\rho_{ji}^W)^2} \quad (2)$$

where, $\vartheta_{ji}^{(W,n)}$ is the mean function that varies based on n , and ρ_{ji}^W is the variance that is set as constant. The formulation of $\vartheta_{ji}^{(W,n)}$ and ρ_{ji}^W can be provided below,

$$\vartheta_{ji}^{(W,n)} = \varpi_{\downarrow}(nei_{ji}^W), \downarrow = 1, 2, 3, \dots, h \quad (3)$$

$$\rho_{ji}^W = \varpi_h(R_{ji}^W) \quad (4)$$

From the above equations, ϖ_{\downarrow} is the mean of \downarrow middle, and ϖ_h is the standard mean. Utilizing ι_1 norm, the R_{ji}^W can be formulated as follows,

$$R_{ji}^W = \{|k_m - av_{\vartheta}|, \forall k_m \in nei_{ji}^W\} \quad (5)$$

$$av_{\vartheta} = \frac{1}{h} \sum_{n=1}^h \vartheta_{ji}^{(W,n)}$$

where, the average mean can be defined as av_{ϑ} from the mean of \downarrow middle. From the $\delta_{mem_{ji}^{(W,n)}}$, mean, and variance the membership matrix (∇_{ji}) is constructed with size of $h \times M$ that composed values of membership function of k_m .

From the matrix, the threshold value (th^n) is determined for pixel classification that can be formulated as,

$$th^n = \min(\max(\nabla_{ji})) \quad (6)$$

From the above equation, \min and \max denotes the minimum and maximum operators respectively. In the ∇_{ji} , the column wise operation includes the association of mem_{ji}^W with nei_{ji}^W that can be expressed as,

$$\delta_{(mem)_{ji}^W} = \frac{\sum_{n=1}^h \nabla_{ji}}{h} \quad \forall j = 1, \dots, M \quad (7)$$

From the Equations (6) and (7), the pixel quality is determined. If the $\delta_{(mem)_{ji}^w} > th^n$ then the pixel is considered as good otherwise considered as noisy pixels.

Stage 2: In this stage, the noisy pixels of the input images were denoised based on the good pixels. The good pixels sets were denoted as β with mapping function of $[0,1]$ by the membership function δ_β . The mean gaussian membership function is computed for the β based on mean of β middle. From which the average of β - values is taken from the set of β . Therefore, the avg_m and ρ_β of δ_β can be formulated as,

$$avg_n = \frac{\sum_{n=1}^h m_n}{h}; \rho_\beta = |\beta - avg_m| \quad (8)$$

$$\delta_\beta(g_j) = e^{-(g_j - (avg)_m)^2 / 2\rho_\beta^2} \quad (9)$$

From the above Equation (9), m_n is the n -th good pixels mean ($n = 1, 2, \dots, h$). The denoise intensity pixels can be formulated as,

$$de_{pix} = \frac{\sum_{\forall g_j \in \beta} \mathbb{I}_j g_j}{L}; L = \sum_{j=1}^{de} I_j \quad (10)$$

where, the weight of the good pixel can be denoted as $I_j \in \delta_\beta$, and the normalized term is denoted as L . The pseudo code for the proposed noise removal step using A-Fuzzy in multi stage pre-processing is provided below.

Algorithm 1 Pseudocode for Noise Removal Using A-Fuzzy

```

1: Input:Noisy 2D Image
2: Output:Denoised Image
3: Begin
4: //Noise Removal//
5: for all input images do
6:   for every  $pix_{ji}$  in  $K$  do
7:     Determine the neighborhood pixels  $nei_{ji}^W$  (1)
8:     Compute gaussian membership function (2)
9:     Determine mean ( $\vartheta_{ji}^{(W,n)}$ ) and variance ( $\rho_{ji}^W$ ) (3), (4)
10:    Construct membership matrix  $\nabla_{ji}$ 
11:    Determine threshold and  $\delta_{(mem)_{ji}^w}$  (6) and (7)
12:    if  $\delta_{(mem)_{ji}^w} > th^n$  then
13:      Good Pixel
14:    else
15:      Bad Pixel
16:    end if
17:  end for
18: end for
19: End

```

4.1.2. Contrast enhancement

After removing the noise, contrast enhancement is performed to improve the quality of RGB-D images using MSO algorithm which equalizes the histogram of the images by improving the visibility based on brightness adjustment in an efficient manner. This equalizes the luminance to increase the detection accuracy.

The noise removed image is denoted as $de(j, i)$ in which $j = 1, 2, \dots, N$ and $i = 1, 2, \dots, R \in z^{N \times R}$ in which the (j, i) is the gray location in the image of size $N \cdot R$. At first, the given denoised image is segmented into non-overlapping segments as $S = \{s_1, s_2, \dots, s_n\}$. Furthermore, the segments are divided into blocks that can be represented as $B = \{b_1, b_2, \dots, b_{n-1}\}$. From that, the Contrastness Measure (CM) is determined. For computing CM, the Contrast Value Factor (CVF) is determined that can be formulated as,

$$CVF(l) = CM_{mw} + CM_{wD}, \quad l \in [1, 2, \dots, n] \quad (11)$$

Where, CM_{mw} is the contrast measure with mean window, and CM_{wD} is the contrast measure with window deviation. Both are computed based on the non-overlapping segments. At last, the contrast score is determined by,

$$con_{sc} = \frac{1}{n} \sum_l^n CVF(l), \quad l \in [1, \dots, n] \quad (12)$$

From the Equation (12), the given image con_{sc} can be determined based on the probability value from high to low contrast. If the given image has lower contrast, then the proposed work utilized MSO algorithm to enhance the contrast based on equalizing the histogram. In our work, the less contrast pixels in the images are considered as moths (q_i) and their histogram is $F(q_i)$. For every iteration, the contrast of pixels is improved. The positions of the less contrast pixels are initialized as follows,

$$q_{iv} = Rnd[0, 1] \cdot (q_v^{maxi} - q_v^{mini}) + q_v^{mini} \quad (13)$$

where, $i \in \{1, 2, \dots, q\}$, $v \in \{1, 2, \dots, d\}$ in which q denotes the pixel population, d is the problem dimension, and Rnd is the random value. Whereas, the q_v^{mini} and q_v^{maxi} are the lower and upper limits respectively. The objective function of moth swarm optimization based contrast enhancement is shown in Equation (12). From that, the probability of updation can be formulated as,

$$Pr_u = \frac{F(q_i)_u}{\sum_{u=1}^{qu} F(q_i)_u} \quad (14)$$

For optimizing the histogram of the pixels for reducing the luminance, the following conditions must be satisfied based on the contrast score (i.e., objective function) that can be formulated as,

$$F(q_i)_u = \begin{cases} \frac{1}{1+con_{sc}}, & con_{sc} \geq 0 \\ 1 + |con_{sc}|, & con_{sc} < 0 \end{cases} \quad (15)$$

The updation of low contrast pixels to high contrast pixels after contrast enhancement can be formulated as,

$$q_i^{j+1} = q_i^j + 0.001 \cdot \alpha [q_i^j - q_i^{mini}, q_i^{maxi} - q_i^j] + (1 - \aleph/\alpha) \cdot Rnd_1 \cdot (bes_u^i - q_i^j) + 2\aleph/\alpha - Rnd_2 \cdot (bes_{\aleph}^i - q_i^j) \quad (16)$$

Where, Rnd_1 and Rnd_2 are the random numbers of interval $[0,1]$. \aleph/α , and $2\aleph/\alpha$ are the environmental factors affecting the contrast enhancement. During the end of current iteration, the contrastness of the pixels are refined for next iteration which iterates until the desired solution had met.

4.1.3. Points to voxel conversion

The enhanced LiDAR 2D cloud points are converted into 3D voxels for improving the perception view of the object to increase the detection accuracy of the 3D objects. For converting the 2D LiDAR to 3D voxels, the maximum and minimum points are traversed in three dimensional directions (i.e., X , Y , and Z). The maximum traversed point is $(maxi_x, maxi_y, maxi_z)$, and the minimum traversed point is $(mini_x, mini_y, mini_z)$. The voxel grid can be computed by rounding operation based on the voxel size that can be formulated as,

$$\left\lceil \frac{maxi_x - mini_x}{VoxS} \right\rceil \cdot \left\lceil \frac{maxi_y - mini_y}{VoxS} \right\rceil \cdot \left\lceil \frac{maxi_z - mini_z}{VoxS} \right\rceil \quad (17)$$

From the obtained voxel grid, the 3D voxel grid coordinates for every point clouds can be determined as,

$$\begin{cases} Vox_{i,X} = \left\lceil \frac{Pt_i.X - mini_x}{VoxS} \right\rceil \\ Vox_{i,Y} = \left\lceil \frac{Pt_i.Y - mini_y}{VoxS} \right\rceil \\ Vox_{i,Z} = \left\lceil \frac{Pt_i.Z - mini_z}{VoxS} \right\rceil \end{cases} \quad Pt_i \in pointcloud \quad (18)$$

From the above equation, Pt_i denotes the i -th point in the 2D LiDAR point cloud. The voxelized image and the contrast enhanced image are fused to form high refined 3D image.

4.2. L-GAN based instance segmentation

The voxelized LiDAR cloud points and pre-processed RGB-D images are fused before performing segmentation for achieving efficient results in detection of 3D objects. The angle of the images is changed from one another which reduces the detection accuracy. To overcome this issue, we rotate the images in terms of several degrees such as 10° , 90° , 180° , and 270° . After rotating the images, instance segmentation

is performed for the fused images using Lightweight Generative Adversarial Network (L-GAN) algorithm which provides precise segmentation when compared with other state-of-the-art models in terms of having channel and position attention modules respectively (Ch_{att} & Po_{att}).

The generator (Gen) in the GAN is trained in the way of mapping function from input image to segmented image. The Gen composed of encoder and decoder architecture. The training of Gen is carried out using loss functions from discriminator (Dis) to Gen . For instance, the input object image is denoted as ‘ a ’ and the ground-truth image is denoted as ‘ b ’. A random variable ‘ E ’ is introduced to reduce the overfitting at the decoder layer. Therefore, outputs of Gen and Dis can be represented as $Gen(a, E)$ and $Dis(a, Gen(a, E))$. With that, the generator loss function can be formulated as,

$$Gen_{loss}(Gen, Dis) = \mathbb{E}_{a,b,E}(-\log(Dis(a, Gen(a, E)))) + \gamma \mathbb{E}_{a,b,E}(L1_{loss}(b, Gen(a, E))) + \varphi \mathbb{E}_{a,b,E}(jacc_{loss}(b, Gen(a, E))) \quad (19)$$

where, γ and φ are the factor of weights. Our work considers three losses such as Jaccard loss, $L1$ loss, and adversarial loss. The reason for adopting three loss functions is that, as the adversarial loss might slow down the learning process so that $L1$ loss is utilized for preserving the object boundaries and Jaccard loss is utilized for improving the relationship among the original and segmented image.

On the other side, the discriminator Dis composed of four layers such as convolutional, position attention, channel attention, and activation layer respectively for robustly finds the generated images into real or fake. The loss function associated with the Dis can be formulated as,

$$Dis_{loss}(Gen, Dis) = \mathbb{E}_{a,b,E}(-\log(Dis(a, Z))) + \mathbb{E}_{a,b,E}(-\log(1 - Dis(a, Gen(a, E)))) \quad (20)$$

From the Dis_{loss} , the loss of binary entropy can be effectively determined by two mathematical terms such as $-\log(Dis(a, Z))$ (i.e., ground-truth image) and $-\log(1 - Dis(a, Gen(a, E)))$ (i.e., predicted image). The optimizer in the Dis performs minimization and maximization of loss function for predicted and ground truth images with classes of 0 and 1 respectively.

The attention modules in the encoder and decoder of Gen is utilized for learning both the high and low level features respectively. The Ch_{att} is utilized for learning the high level features by learning the feature interdependencies. From the features $\cup \in \mathbb{K}^{(C \cdot He \cdot Wi)}$, the Ch_{att} generates the channel attention map $\mathbb{X} \in \mathbb{K}^{C \cdot C}$ in which the C , He , and Wi denotes the channel, height, and width of the given image.

Utilizing softmax function, the $\mathbb{X} \in \mathbb{K}^{(C \cdot C)}$ is created as follows,

$$y_{ji} = \frac{\exp(U_i \cdot U_j)}{\sum_{i=1}^C \exp(U_i \cdot U_j)} \quad (21)$$

From the above equation, $U_i.U_j$ denotes the transpose of matrix multiplication, y_{ji} denotes the impact of i -th channel on j -th channel. The multiplicative results are reshaped to $\mathbb{K}^{(C \cdot He \cdot Wi)}$ that is again multiplied by χ (a scalar parameters).

After that, element wise addition is undergone to provide the output is $E \in \mathbb{K}^{(C \cdot He \cdot Wi)}$ as,

$$E_j = \chi \sum_{i=1}^C (y_{ji} U_i) + U_j \quad (22)$$

The final feature representation is the sum of weights of features of all channels which can provides the semantic dependencies and enhance the decimator functions.

Once, the important features are obtained from the Ch_{att} , the contextual information are obtained by the Po_{att} . In simple words, the Po_{att} encodes the contextual information to local features and represents them to local feature maps $\in \mathbb{K}^{(C \cdot He \cdot Wi)}$.

The feature maps are then provided to the consecutive convolutional layers for generating the other two feature maps that is represented as $(B, C) \in \mathbb{K}^{C \cdot He \cdot Wi}$.

The feature maps are then reshaped and fed to softmax layer for generating the spatial feature map that can be formulated as,

$$SP_{ji} = \frac{\exp(B_i, C_j)}{\sum_{i=1}^N \exp(B_i, C_j)} \quad (23)$$

Where, Sp_{ji} refers to j -th spatial position interaction on i -th position. The association among the feature maps is ensured by the softmax layer. For instance, the U is provided to the convolutional layers for generating a new feature map $D \in \mathbb{K}^{C \cdot N}$.

The output from the Po_{att} can be computed by multiplying the transpose of Sp_{ji} and D that can be formulated as:

$$E_j = \xi \sum_{i=1}^N (Sp_{ji} D_i) + U_j \quad (24)$$

Where, the scalar constraint is represented as ξ . The Po_{att} output is sum of weight of neighbor features which represents the context information of local features through spatial map representation.

Figure 2 represents the diagrammatic view of L-GAN based instance segmentation of objects.

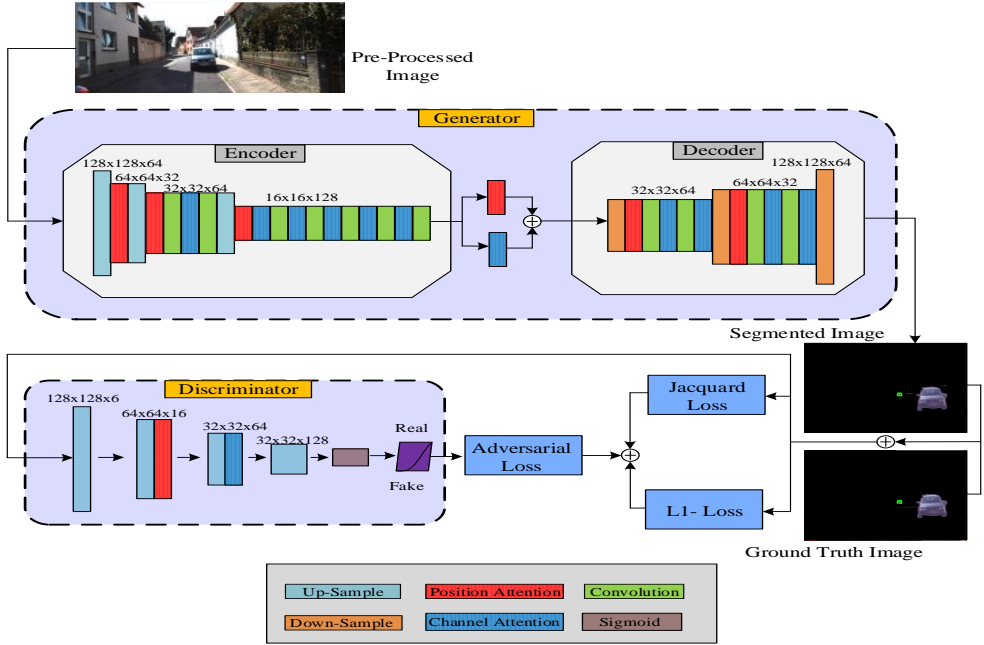


Figure 2. L-GAN based Instance Segmentation

4.3. 3D object detection and tracking

After segmenting the images, we perform feature extraction from the segmented region for classification of objects. Feature extraction and classification are performed using Hybrid Deep Learning algorithm which consists of YOLOv4 and VGG16 algorithms. YOLOv4 is mainly implemented to increase the classification speed and detect small objects. VGG16 is implemented to increase detection accuracy. In this proposed, we extract numerous features such as spatial, temporal, textural, visual, and auditory features using VGG16, and classification is performed using YOLOv4 which provides four classes such as ground, vehicles, pedestrians, and obstacles. Figure 3 represents the process of 3D object detection.

4.3.1. Feature extraction-VGG 16

The segmented image from the LGAN of input size 224×224 is provided with R, G, and B channels. The input size of the image is reduced for every pixel for achieving the desired results. Once, the images are passed over the ReLU activations, the resultant image is provided to the stack of two consecutive convolutional layers with area size of 3×3 and have 64 filters. The image is processed at 1 pixel padding and convolutional stride is also at 1 pixel. The two consecutive layer preserves the spatial resolution with pooling of two pixel of window size 2×2 , so that the activation window size is reduced to half.

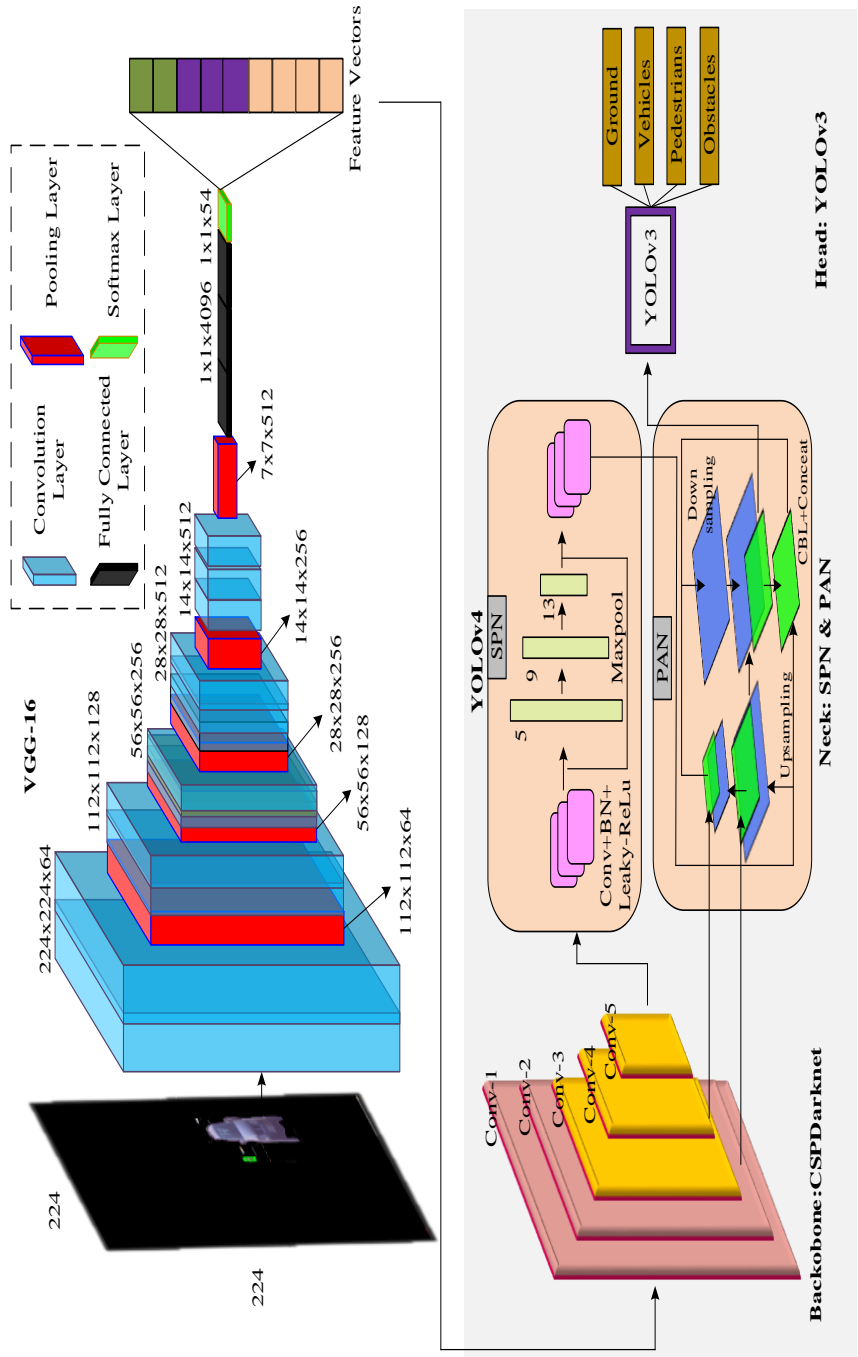


Figure 3. VGG-16 & YOLOv4 based Feature Extraction and Classification

The output activation function from the first stack of convolution is then provided to second stack of activation and convolutional layers in which the activation layer 128 filters with size of $56 \times 56 \times 128$, and three convolutional layers has 256 filter with size of $56 \times 56 \times 256$. In similar manner, the filter in the consecutive three layers is increased to 512 and convolution layer size is reduced to of $28 \times 28 \times 512$, $14 \times 14 \times 512$, and $7 \times 7 \times 512$. The output images from the final max pooling layer are then provided to the 3 fully connected layer in which the first two fully connected layers composed of 4096 channels and last one layer has 1000 channels. Finally, the softmax layer provides the feature vectors of the segmented images that includes feature representation vectors of features such as spatial, temporal, textural, visual, and auditory features which can be represented as,

$$Fe = \begin{bmatrix} Fe_1 \\ Fe_2 \\ Fe_3 \\ Fe_4 \\ Fe_5 \end{bmatrix} \quad (25)$$

where, Fe_1 , Fe_2 , Fe_3 , Fe_4 and Fe_5 represents the features such as spatial, temporal, textural, visual, and auditory respectively.

4.3.2. Object classification – YOLOv4

The extracted feature vectors are then provided to the YOLOv4. The YOLOv4 is developed by improving the YOLOv3 form improving the speed, detailed, and stable results. The YOLOv4 composed of backbone network, neck, and head for processing the input features and provides the desirable output. To be clearer, the proposed work used backbone network as Cross Scale Partial Darknet-53 (CSPDarknet-53), the neck structure used are Spatial Pyramidal Pooling (SPN) and Path Aggregation Network (PAN), the head structure has YOLOv3 for enabling speedy object classification.

The extracted feature vectors are provided to the CSPDarknet-53 for representing the deep features using five ResNet blocks. The ResNet blocks composed of fifty-three convolutional layers of sizes 3×3 and 1×1 with connection to batch normalization, and mesh activation layer respectively. For reducing the computation complexity, the conventional ReLU is substituted with the leaky ReLU. The represented features from the CSPDarknet-53 are then provided to the neck that consist of SPN and PAN. The SPN composed of several maxpooling layers of different sizes such as 13, 9, and 5 to normalize the features sizes through cross minibatch normalization. The normalizes features are provided to the PAN for continuously extracting the features in repeated fashion in top down and bottom down approach. The extracted deep features, are finally provided to the YOLO head. The proposed YOLO head utilized YOLOv3 for object detection with the size of 76×76 to detect and classify the object of varying sizes.

The classification result can be represented as:

$$YOLOv4_{Head} = \begin{cases} Ground \\ Vehicles \\ Pedestrains \\ Obstacles \end{cases} \quad (26)$$

Finally, loss of the YOLOv4 is computed which consist of three losses such as object classification, localization, and offset loss respectively. The formulation of loss function is computed below,

$$L^{oss} = \Xi_1 L^{cl} + \Xi_2 L^{loc} + \Xi_3 L^{con} \quad (27)$$

Where, L^{cl} , L^{loc} , and L^{con} states the classification, localization, and confidence loss respectively. The Ξ_1 , Ξ_2 , and Ξ_3 are the balancing factors of the respective loss functions. The formulation of individual loss functions can be formulated as,

$$L^{cl} = -\sum_{i \in box} \sum_{j \in class} (ob_{ij} \ln(pr_{ij}) + (1 - ob_{ij}) \ln(1 - pr_{ij})) \quad (28)$$

$$L^{loc} = 1 - InOU(Pre, GnT) + \frac{d_{Pre, GnT}^2(Pre_{cen}, GnT_{cen})}{l^2} + \delta \quad (29)$$

$$L^{con} = -\sum (ob_i \ln(pr_i) + (1 - ob_i) \ln(1 - pr_i)) \quad (30)$$

From the Equation (28), pr_{ij} and ob_{ij} represents the i -th object class in the boundary box prediction i . From Equation (29), InOU is the intersection over union, Pre , GnT denotes the predicted and ground truth results respectively, Pre_{cen} , GnT_{cen} defines the center point euclidean distance, and δ denotes the facet ratio. From the Equation (30), ob_i represents whether there is any object in the bounding box $[0,1]$, and pr_{ij} probability of the object in the bounding box.

4.3.3. Object tracking-IKF

After classification of objects, tracking is performed only for moving objects by considering RFID, unique ID, dimension, and orientation using Improved Unscented Kalman Filter (IUKF) which reduces the variance and tracks the objects with high accuracy by considering the object's velocity and location. Time-based mapping is performed by considering previous and current time, location from the RFID to increase the tracking reliability.

At a 3D plane, let us consider the detected object at the previous stage is moving at a uniform speed. The state of the moving object is denoted as $d[u]$ with time u . The position of the object in 3D plane time u is denoted as $pos_x[u]$, $pos_y[u]$, $pos_z[u]$. The detected bound box aspect ratio $asp[u]$, height $hei[u]$, object velocity

$vel_x[u], vel_y[u], vel_z[u]$, location of the object $loc[u]$, and its unique id $obj[ID_u]$. The complete details are denoted as,

$$d[u] = (pos_x[u], pos_y[u], pos_z[u], vel_x[u], vel_y[u], vel_z[u], asp[u], hei[u], loc[u], obj[ID_u])^T \in \wedge^{10} \quad (31)$$

The state of an object at time $u + 1$ can be formulated as,

$$d[u + 1] = Fd[u] + Ng\Psi[u] \quad (32)$$

Where, F denotes the transfer matrix for the previous object state, Ng is the noise matrix, and $\Psi[u]$ represents the noise vector of a system at time u . Based on the mentioned object motion model, the IUKF algorithm is utilized for object state tracking. At first, the sigma points group are constructed as,

$$\Gamma_i = \begin{cases} \bar{d}[u] + (\sqrt{(dim_{sv} + \Upsilon)err[u]}), i = 1, 2, \dots, L \\ \bar{d}[u] - (\sqrt{(dim_{sv} + \Upsilon)err[u]}), i = L + 1, \dots, 2L \\ \bar{d}[u], i = 0 \end{cases} \quad (33)$$

Where, $err[u]$ represents the covariance error matrix, dim_{sv} is the state vector dimension, and Υ is the distance parameter of sigma points. The sigma points are substituted to the non-linear equation that can be formulated as,

$$y_i = h(\Gamma_i), i = 0, 1, \dots, 2dim_{sv} \quad (34)$$

From the y , mean and variance are computed that can be formulated as follows,

$$\bar{y} \approx \sum_{i=0}^{2dim_{sv}} W_i^{(m)} y_i \quad (35)$$

$$err_{\Upsilon} \approx \sum_{i=0}^{2dim_{sv}} W_i^{(c)} (y_i - \bar{y})(y_i - \bar{y})^T \quad (36)$$

The computation of $W_i^{(m)}$, and $W_i^{(c)}$ is provided as follows,

$$W_0^{(m)} = u / (dim_{sv} + u) \quad (37)$$

$$W_0^{(c)} = u / (dim_{sv} + u) + (1 + \tau^2) \quad (38)$$

$$W_i^{(m)} = W_i^{(c)} = u / [2(dim_{sv} + u)], i = 1, \dots, 2dim_{sv} \quad (39)$$

In order to regularize the UKF, the correctness factor Γ^* is introduced to improve the UKF. The adoption of Γ^* reduces the chance of wrong measurements during object tracking. The formulation of Γ^* in the proposed IUKF can be provided as,

$$\Gamma^* = \bar{d}[u] + u(y_u - \hat{y}_{\bar{u}}) \quad (40)$$

Once the correction is completed, state of the object can be formulated as,

$$f = \min(\bar{y}_{u+1} - \bar{y}_{u+1})^T (\bar{y}_{u+1} - \bar{y}_{u+1}), obl_m \leq \tau \leq ob_{up} \quad (41)$$

where, obl_m is the lower limit of the moving object, and ob_{up} is the upper limit of the moving object. The tracking continues until the maximum number of steps $(u + 1)$. The pseudocode denotes the IUKF based 3D object tracking.

Algorithm 2 Pseudocode for IUKF Object Tracking

```

1: Input:Detected Object with Bounding Box
2: Output:Object Tracking
3: Begin
4: Initialize the object tracking model (31)
5: Formulate the object consecutive states (32)
6: //IUKF based Object Tracking//
7: for all detected objects do
8:   Construct the sigma points  $\Gamma_i$  (33)
9:   Compute the  $y_i \rightarrow h(.)$  (34)
10:  Compute mean and variance (35)–(36)
11:  Compute  $W_i^{(m)}$  and  $W_i^{(c)}$  (37)–(38)
12:  Regularize  $UKF \rightarrow \Gamma^*$  (40)
13:  Obtain object state (41)
14:  Track until (u+1)
15: end for
16: End

```

Pseudocode explanation

Step 1: The Object detected with bounding boxes will be given as input.

Step 2: Initialize the object tracking model in 3D Plane with the complete details of Position pos, velocity vel, aspect ratio asp, height hei, location loc, unique id obj[ID].

Step 3: The State of an object with time $u + 1$ will be formulated with noise matrix Ng, transfer matrix from the previous state F, noise vector $\Psi[u]$ at time u.

Step 4: After all the objects has been detected, construct the sigma points group with error matrix err[u], dimension and the distance.

Step 5: Substitute the values of sigma points to non-linear equation Y_i .

Step 6: From Y_i , Calculate mean and variance \bar{y} and err_y

Step 7: Compute the value of $W_i^{(m)}$, and $W_i^{(c)}$, which is calculated during the findings of mean and variance.

Step 8: The correctness factor Γ^* is given to reduce the chance of wrong measurements during tracking.

Step 9: After Correction, state of object can be formulated.

Step 10: Tracking continues until the maximum number of steps reached $(u + 1)$.

5. Experimental results

This section provides the detailed view of simulation, implementation, and comparative results. For diminishing the readers difficulty, separate sections are provided for simulation and implementation results, dataset description, comparative results, and summary.

5.1. Simulation setup

The proposed Hybrid Deep Learning based Multi Object Detection and Tracking (HDL-MODT) is simulated using MATLAB tool of version R2020a. The simulation results show that, the proposed work outperforms than the existing work. The proposed simulation is packaged with pre-processing, segmentation, classification, and tracking. To achieve better performance, some of the system configurations must be adjusted. The adjusted system configurations are mentioned in Table 2. Furthermore, the simulation results of the proposed work also provided in the Figure 4a–4d.

Table 2
System configurations

Hardware Configuration	RAM	500GB
	Hard Disk	8GB
Software Configuration	Simulation Tool	MATLAB R2020a
	OS	Windows-10(64-bit) OS
	Processor	Intel(R) Core(TM) i5-4590S CPU@3.00GHz

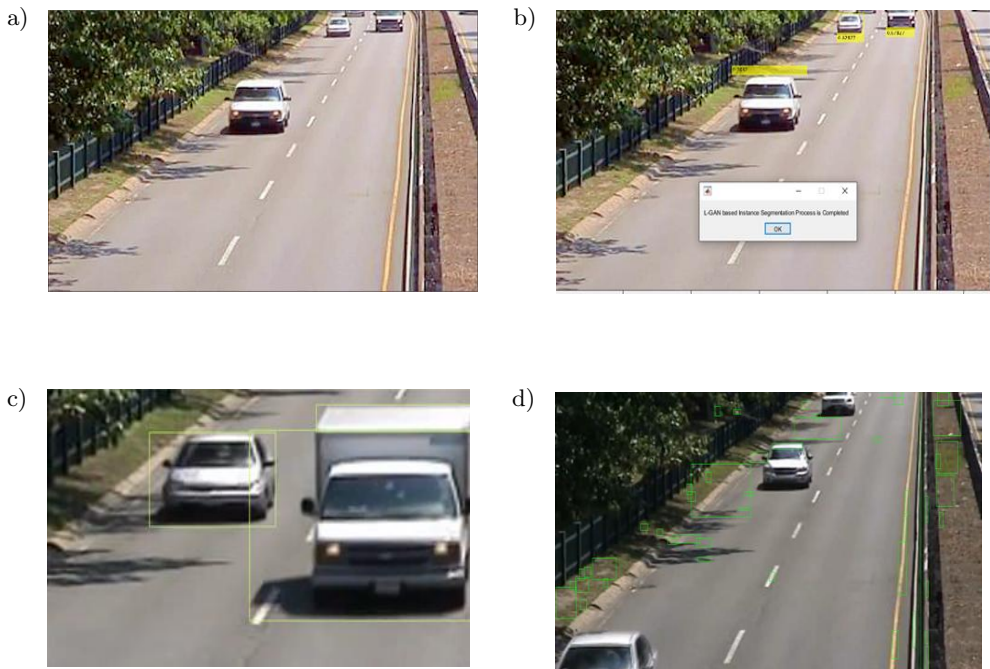


Figure 4. Multi Stage Noise Removal (a); L-GAN based Instance Segmentation (b); Moving Object Detection & Tracking (c); Static Object Detection (d)

5.2. Dataset description

The performance of the proposed work is evaluated by performing quantitative and qualitative experiments using Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) dataset. This dataset [11] holds the images from 3D LiDAR and stereo RGB camera from the autonomous vehicles. The capturing of LiDAR frames by using an LiDAR sensor named HDL-64E. The points generated by the sensor is one million. The frames provided for testing and training is 7518 and 7481 respectively. The proposed work divides the dataset as 25% for validation and 75% for training. The dataset contains 52,979 labels with nine categories such as ‘don’t care objects’, ‘miscellaneous’, ‘tram’, ‘sitting person’, ‘truck’, ‘van’, ‘cyclist’, ‘pedestrian’, and ‘car’.

5.3. Comparative analysis

This sub-section explains the comparative results proposed HDL-MODT with existing works such as PointTrackNet [26], and STR-ODT [12] respectively. The validation metrics taken such as accuracy, precision, recall, f-score, and computation. The brief explanation of the proposed comparative results are defined below.

5.3.1. Accuracy comparison

Accuracy is defined as the sum of True Positive (TP) and True Negative (TN) to the ratio of sum of TP , TN , False Positive (FP), and False Negative (FN) respectively. The formulation of accuracy is,

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (42)$$

Figure 5 represents the comparison of accuracy of proposed and existing works with respect to number of frames.

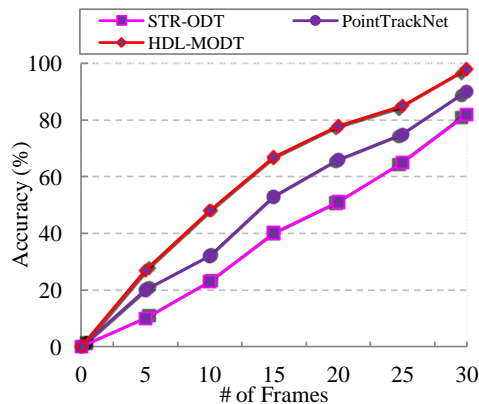


Figure 5. Number of Frames vs Accuracy

From the inference, it is shown that when the number of frames increases the accuracy rate also increases. The reason for such higher increment in accuracy is that, the proposed work utilized hybrid deep learning algorithm named VGG-16 and YOLOv4 for feature extraction and classification respectively. On contrary, the existing work PointTrackNet lacks with extracting optimal features and poor classifier for object detection leads to less accuracy. On the whole, the graphical inference shows that, our proposed work achieves higher accuracy than the existing works.

The numerical results show that, the proposed work achieves higher accuracy of 98% when the frames increased to 30 whereas the existing works PointTrackNet and STR-ODT achieves lesser accuracy of 90% and 82% respectively. Overall, the proposed work achieves higher accuracy of 8–16% than the existing works.

5.3.2. Precision comparison

The precision is defined as the ratio of TP to the sum of TP and FP respectively. In other words, its also define how precisely the proposed work classifies and tracks the objects. The formulation of precision is provided as below,

$$Pre = \frac{TP}{TP + FP} \quad (43)$$

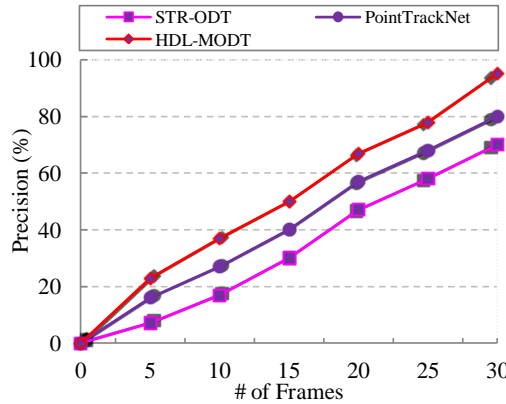


Figure 6. Number of frames vs precision

Figure 6 shows the comparison precision with proposed and existing in terms of number of frames. The precision rate increased with increase in number of frames. Among that our proposed work achieves higher precision than the existing works. The reason for such higher precision rate is that, the proposed work performs improved deep learning based segmentation named LGAN based instance segmentation, and IUKF based object tracking. In IUKF based objects, the detected moving objects are tracked by considering metrics such as velocity, location, RFID, dimension, and

unique ID. Furthermore, the proposed work also utilized time-based mapping to precisely track down the objects. The existing work lacks with less precision rate, as they were not performing segmentation which increase the higher false positive rates. In addition to that, the existing object tracking feature was not plausible that also affects the precision in object tracking.

The numerical results show that, the proposed work achieves higher precision of 95% when the frames increased to 30 whereas the existing works PointTrackNet and STR-ODT achieves lesser precision of 80% and 70% respectively. Overall, the proposed work achieves higher precision of 5–25% than the existing works.

5.3.3. Recall comparison

The recall rate is defined as the ratio of TP to the sum of TP and FN respectively. The proposed work defines the recall rate by computing the amount of positively detected samples. The formulation of proposed recall rate is as follows,

$$Rec = \frac{TP}{TP + FN} \quad (44)$$

The comparison of recall rate with respect to number of frames for the proposed and existing works is shown in Figure 7. The figure shows that, the recall rate increases with increase in frame rate. The major reason for such higher recall rate is that, the proposed work performs multi stage pre-processing method and thereby the rate of correctly classifying the samples is increased. The proposed pre-processing method firmly increases the performance of accuracy and precision respectively. The existing works PointTrackNet and STR-ODT limits with pre-processing of acquired images thereby they achieve deprived performance on further processes. Hence, the probability of positively classifying the samples is less in the existing works.

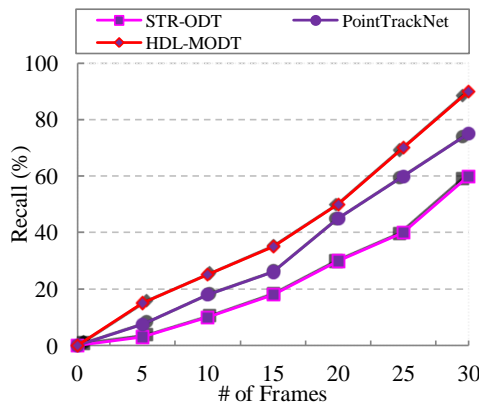


Figure 7. Number of frames vs Recall

The numerical results show that, the proposed work achieves higher recall rate of 90% when the frames increased to 30 whereas the existing works PointTrackNet and STR-ODT achieves lesser recall rate of 75% and 60% respectively. Overall, the proposed work achieves higher recall rate of 20–30% than the existing works.

5.3.4. F-Score comparison

The *F-Score* is defined as the harmonic mean of precision and recall rates respectively. To be clear, the mathematical illustration of *F-Score* can be formulated as,

$$F\text{-Score} = 2 \cdot \frac{Pre \cdot Rec}{Pre + Rec} \quad (45)$$

The comparison of *F-Score* rate of proposed and existing works with respect to number of frames is shown in Figure 8. The figure shows that when the number of frames increases the *F-Score* rate also increases. From which the proposed work achieves higher *F-Score* rate than the existing works. As the proposed work performs effective pre-processing, and segmentation respectively. The proposed work adopts L-GAN for segmenting the objects in which it performs instance segmentation to merely classify the objects to reduce the unwanted discrepancies during classification thereby improving the *F-Score* rate. In contrast, the existing works PointTrackNet and STR-ODT achieves less *F-Score* rate as they lack with pre-processing by directly provides the images for further process, and also performs ineffective segmentation which reduced the *F-Score* rate.

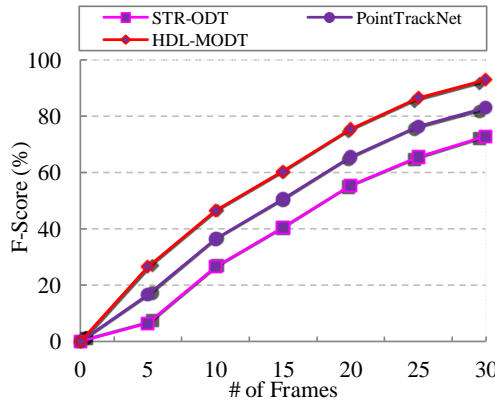


Figure 8. Number of frames vs *F-Score*

The numerical results show that, the proposed work achieves higher *F-Score* rate of 93% when the frames increased to 30 whereas the existing works PointTrackNet and STR-ODT achieves lesser *F-Score* rate of 83% and 73% respectively. Overall, the proposed work achieves higher *F-Score* rate of 10–20% than the existing works.

5.3.5. Computation time

The computation time is defined as the amount of time taken to complete as process. The mathematical formulation of computation time is defined as the ratio of overall computation time to the time taken for computation,

$$CT = \frac{CT_{Time}}{Ov_{Time}} \quad (46)$$

where, CT_{Time} is the computation time taken, and Ov_{Time} is the overall computation time.

Figure 9 shows the comparison of computation time of proposed and existing works with respect to number of frames respectively. From the graphical inference, the computation time of proposed work decreases with increase in number of frames. The reason for such less computation time is that, the proposed work adopts multi stage pre-processing and hybrid deep learning based object detection respectively. The object detection is performed using hybrid deep learning algorithm named VGG-16 and YOLOv4 respectively. The proposed process reduces those complexity by increasing the computation time. On the other hand, the existing works PointTrackNet and STR-ODT gains with higher computation time as they lack with effective pre-processing and classification respectively thereby time for computation was high.

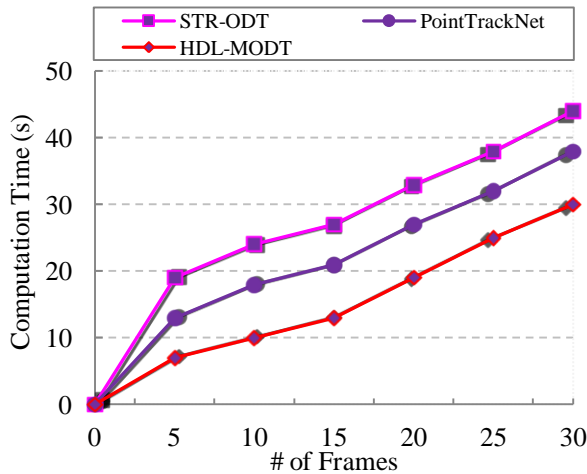


Figure 9. Number of frames vs computation time

The numerical results show that, the proposed work achieves lesser computation time of 30 s when the frames increased to 30 whereas the existing works PointTrackNet and STR-ODT achieves higher computation time of 38 s and 44 s respectively. Overall, the proposed work achieves lesser computation time of 8–14 s than the existing works.

5.4. Research summary

The summary of the experimental results section is provided in this supplementary section. The proposed work composed of sequential processes such as multi stage pre-processing, instance segmentation, and 3D object detection & tracking. The simulation of proposed work is carried out using MATLAB R2020a in which the simulation results are provided in Figures 4a–d. The comparative results of the proposed work and existing works in various simulation metrics are shown in Figures 5–9. The average simulation results comparison of proposed and existing also shown in Table 3. Some of the major highlights of the proposed work are given below:

- For enhancing the quality of the images (i.e., solid-state LiDAR, pseudo-LiDAR, and RGB-D), we perform multi-stage preprocessing in which noise is removed by A-Fuzzy filter and contrast enhancement using MSO algorithm. In addition, point to voxel conversion is performed for achieving efficient object detection results.
- For improving the viewport of the image by performing image rotation and instance segmentation. The L-GAN algorithm is implemented to perform instance segmentation that maximizes the accuracy of object detection which increases the reliability of tracking.
- For increasing the accuracy and speed of 3D object detection, we perform multiple feature extraction and classification by Hybrid deep learning algorithm which detects the objects with low false positive rate and high speed.
- For increasing the tracking reliability, we implement IUKF filter by considering numerous metrics and performing time-based mapping which increases the reliability of tracking with high accuracy.

Table 3
Average comparison of proposed vs existing

Metrics	HDL-MODT	PointTrackNet	STR-ODT
Accuracy [%]	57.57	48	38.72
Precision [%]	50	41.142	32.71
Recall [%]	40.71	33.072	23
F-Score [%]	55.5	46.93	38.22
Computation Time [s]	14.85	21.28	26.42

6. Conclusion

High false positive rates, high computation time, and less QoS are the major issues in the 3D object detection and localization. So that, we tend to resolve that issue by proposing HDL-MODT method. The proposed work adopts KITTI dataset for training and testing the classifiers. Initially, the images captured from the RGB-D cameras

and Solid-State LiDAR are pre-processed in multi stages. The proposed work performs three stages of pre-processing such as noise removal using A-Fuzzy, contrast enhancement using MSO, and point to voxel conversion respectively. The pre-processed image is fused to improve the image quality. Secondly, the fused image is provided for instance segmentation using L-GAN in which position and channel attention are adopted for segmenting the possible objects in the input images. The fused images are then provided for object detection, classification, and tracking. The VGG-16 is utilized for feature extraction which extracts the optimal features such as spatial, temporal, textural, visual, and auditory features. The extracted features are represented in form of feature vectors. The feature vectors are provided as an input to the YOLOv4 classifier for object detection and classification task which classifies the objects into four classes such as ground, vehicles, pedestrians, and obstacles and two categories as static and moving objects. For the moving objects, we perform tracking using IUKF algorithm based on metrics such as RFID, unique ID, location, dimension, and velocity. The time based mapping is also performed to enhance the tracking accuracy. The simulation of proposed work is carried out using MATLAB R2020a simulation tool and performance of the proposed work is validated by considering metrics such as accuracy, precision, recall, F-score, and computation time.

Acknowledgements

The authors thank the reviewer(s) for their scholarly comments and suggestions. The authors also express their gratitude to the Editor-in-Chief (Jacek Kitowski), the Editor, and the Editorial Office Assistant(s) of this journal for managing this manuscript.

References

- [1] Bai J., Li S., Huang L., Chen H.: Robust detection and tracking method for moving object based on radar and camera data fusion, *IEEE Sensors Journal*, vol. 21(9), pp. 10761–10774, 2021. doi: 10.1109/jsen.2021.3049449.
- [2] Bashar M., Islam S., Hussain K.K., Hasan M.B., Ashikur Rahman A.B.M., Kabir M.H.: Multiple object tracking in recent times: A literature review, *arXiv preprint arXiv:220904796*, 2022. doi: 10.48550/arXiv.2209.04796.
- [3] Bescos B., Campos C., Tardós J.D., Neira J.: DynaSLAM II: Tightly-coupled multi-object tracking and SLAM, *IEEE Robotics and Automation Letters*, vol. 6(3), pp. 5191–5198, 2021. doi: 10.1109/lra.2021.3068640.
- [4] Chiu H.K., Li J., Ambruş R., Bohg J.: Probabilistic 3D multi-modal, multi-object tracking for autonomous driving. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14227–14233, IEEE, 2021. doi: 10.1109/icra48506.2021.9561754.
- [5] Choi H., Jeong J., Choi J.Y.: Rotation-Aware 3D Vehicle Detection from Point Cloud, *IEEE Access*, vol. 9, pp. 99276–99286, 2021. doi: 10.1109/access.2021.3095525.

- [6] Fan Y.C., Yelamandala C.M., Chen T.W., Huang C.J.: Real-Time Object Detection for LiDAR Based on LS-R-YOLOv4 Neural Network, *Journal of Sensors*, vol. 2021, pp. 1–11, 2021. doi: 10.1155/2021/5576262.
- [7] Farag W.: Kalman-filter-based sensor fusion applied to road-objects detection and tracking for autonomous vehicles, *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 235(7), pp. 1125–1138, 2021. doi: 10.1177/0959651820975523.
- [8] Huang C., He T., Ren H., Wang W., Lin B., Cai D.: OBMO: One bounding box multiple objects for monocular 3D object detection, *IEEE Transactions on Image Processing*, vol. 32, pp. 6570–6581, 2023. doi: 10.1109/tip.2023.3333225.
- [9] Jiang P., Ergu D., Liu F., Cai Y., Ma B.: A Review of Yolo algorithm developments, *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022. doi: 10.1016/j.procs.2022.01.135.
- [10] Kim A., Ošep A., Leal-Taixé L.: EagerMOT: 3D Multi-Object Tracking via Sensor Fusion, *CoRR*, vol. abs/2104.14682, 2021. doi: 10.1109/icra48506.2021.9562072. 2104.14682.
- [11] KITTI DataSet, <https://universe.roboflow.com/sebastian-krauss/kitti-9amcz/DATASET/2>.
- [12] Koh J., Kim J., Yoo J.H., Kim Y., Kum D., Choi J.W.: Joint 3D object detection and tracking using spatio-temporal representation of camera image and LiDAR point clouds. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 1210–1218, 2022. doi: 10.1609/aaai.v36i1.20007.
- [13] Lee E., Nam M., Lee H.: Tab2vox: CNN-based multivariate multilevel demand forecasting framework by tabular-to-voxel image conversion, *Sustainability*, vol. 14(18), 11745, 2022. doi: 10.3390/su141811745.
- [14] Liu Z., Cai Y., Wang H., Chen L., Gao H., Jia Y., Li Y.: Robust target recognition and tracking of self-driving cars with radar and camera information fusion under severe weather conditions, *IEEE Transactions on Intelligent Transportation Systems*, vol. 23(7), pp. 6640–6653, 2021. doi: 10.1109/tits.2021.3059674.
- [15] Luo C., Yang X., Yuille A.: Exploring simple 3D multi-object tracking for autonomous driving. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10488–10497, 2021. doi: 10.1109/iccv48922.2021.01032.
- [16] Nabati R., Harris L., Qi H.: CFTrack: Center-based radar and camera fusion for 3D multi-object tracking. In: *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, pp. 243–248, IEEE, 2021. doi: 10.1109/ivworkshops54471.2021.9669223.
- [17] Pal S.K., Pramanik A., Maiti J., Mitra P.: Deep learning in multi-object detection and tracking: state of the art, *Applied Intelligence*, vol. 51, pp. 6400–6429, 2021. doi: 10.1007/s10489-021-02293-7.

- [18] Pang Z., Li Z., Wang N.: SimpleTrack: Understanding and rethinking 3D multi-object tracking. In: L. Karlinsky, T. Michaeli, K. Nishino (eds.), *Computer Vision – ECCV 2022 Workshops. Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, pp. 680–696, Springer, 2022. doi: 10.1007/978-3-031-25056-9_43.
- [19] Park D., Ambruş R., Guizilini V., Li J., Gaidon A.: Is pseudo-lidar needed for monocular 3D object detection? In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3142–3152, 2021. doi: 10.1109/iccv48922.2021.00313.
- [20] Premachandra C., Ueda S., Suzuki Y.: Detection and tracking of moving objects at road intersections using a 360-degree camera for driver assistance and automated driving, *IEEE Access*, vol. 8, pp. 135652–135660, 2020. doi: 10.1109/access.2020.3011430.
- [21] Qian R., Lai X., Li X.: 3D object detection for autonomous driving: A survey, *Pattern Recognition*, vol. 130, 108796, 2022. doi: 10.1016/j.patcog.2022.108796.
- [22] Shreyas E., Sheth M.H., Mohana: 3D object detection and tracking methods using deep learning for computer vision applications. In: *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, pp. 735–738, IEEE, 2021. doi: 10.1109/rteict52294.2021.9573964.
- [23] Simonelli A., Bulò S.R., Porzi L., Kotschieder P., Ricci E.: Are we Missing Confidence in Pseudo-LiDAR Methods for Monocular 3D Object Detection? In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3205–3213, 2021. doi: 10.1109/iccv48922.2021.00321.
- [24] Wang B., Zhu M., Lu Y., Wang J., Gao W., Wei H.: Real-time 3D object detection from point cloud through foreground segmentation, *IEEE Access*, vol. 9, pp. 84886–84898, 2021. doi: 10.1109/access.2021.3087179.
- [25] Wang K., Liu M.: YOLOv3-MT: A YOLOv3 using multi-target tracking for vehicle visual detection, *Applied Intelligence*, vol. 52(2), pp. 2070–2091, 2022. doi: 10.1007/s10489-021-02491-3.
- [26] Wang S., Sun Y., Liu C., Liu M.: PointTrackNet: An End-to-End Network for 3-D Object Detection and Tracking From Point Clouds, *IEEE Robotics and Automation Letters*, vol. 5(2), pp. 3206–3212, 2020. doi: 10.1109/lra.2020.2974392.
- [27] Wang Y., Guizilini V.C., Zhang T., Wang Y., Zhao H., Solomon J.: DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In: A. Faust, D. Hsu, G. Neumann (eds.), *Conference on Robot Learning, 8–11 November 2021, London, UK*, Proceedings of Machine Learning Research, vol. 164, pp. 180–191, PMLR, 2022. <https://proceedings.mlr.press/v164/wang22b.html>.
- [28] Wang Y., Wang C., Long P., Gu Y., Li W.: Recent advances in 3D object detection based on RGB-D: A survey, *Displays*, vol. 70, 102077, 2021. doi: 10.1016/j.displa.2021.102077.
- [29] Wang Y., Yang B., Hu R., Liang M., Urtasun R.: PLUMENet: Efficient 3D object detection from stereo images. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3383–3390, IEEE, 2021. doi: 10.1109/iro51168.2021.9635875.

- [30] Wen L.H., Jo K.H.: Fast and accurate 3D object detection for lidar-camera-based autonomous vehicles using one shared voxel-based backbone, *IEEE Access*, vol. 9, pp. 22080–22089, 2021. doi: 10.1109/access.2021.3055491.
- [31] Xie X., Cheng G., Wang J., Yao X., Han J.: Oriented R-CNN for object detection. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3520–3529, 2021. doi: 10.1109/iccv48922.2021.00350.
- [32] Zhao X., Sun P., Xu Z., Min H., Yu H.: Fusion of 3D LIDAR and camera data for object detection in autonomous vehicle applications, *IEEE Sensors Journal*, vol. 20(9), pp. 4901–4913, 2020. doi: 10.1109/jsen.2020.2966034.

Affiliations

Dheepika PS

Nehru Memorial College (Affiliated to Bharathidasan University), Department of Computer Science, Tiruchirapalli 621007, India, psdheepika@gmail.com

Umadevi V

Nehru Memorial College (Affiliated to Bharathidasan University), Department of Computer Science, Tiruchirapalli 621007, India, yazh1999@gmail.com

Received: 07.07.2023

Revised: 26.04.2024

Accepted: 26.04.2024

HEBA F. EID
ERIK CUEVAS

ENHANCED BONOBO OPTIMIZER FOR OPTIMIZING DYNAMIC PHOTOVOLTAIC MODELS

Abstract *Bonobo optimizer (BO) is a novel metaheuristic algorithm motivated by the social behaviour of the bonobos. This paper presents a quantum behaved bonobo optimization algorithm (QBOA) employing an innovative metaheuristic based on the reproductive strategies and social behavior of bonobos. Whereby, the quantum mechanics are embedded into the bonobo optimizer to direct the search agents through the search space. Accordingly, under this quantum-behaved movement, the proposed QBOA's exploitation capability is promoted. The performance of the proposed QBOA is exhibited on CEC2005 and CEC2019 benchmarks. Moreover, the QBOA algorithm was adapted to optimize the dynamic photovoltaic models parameters. QBOA exhibits the efficiency and adequacy to solve various optimization problems based on experimental and comparison findings, as well as its ability to implement competitive and promising results optimizing dynamic photovoltaic models.*

Keywords bonobo optimization algorithm, quantum behaved mechanism, photovoltaic model, dynamic photovoltaic model, parameters estimation

Citation Computer Science 25(3) 2024: 469–493

Copyright © 2024 Author(s). This is an open access publication, which can be used, distributed and reproduced in any medium according to the Creative Commons CC-BY 4.0 License.

1. Introduction

Meta-heuristics algorithms have been effectively employed to solve a variety of real-world optimization problems, due to their reliable, robust and computationally efficient in avoiding local minima [11]. Various well-known meta-heuristics algorithms that have been proposed in the literature are Particle swarms [10], Coyote Optimization Algorithm (COA) [24], Dragonfly Algorithm (DA) [21], Particle Swarm Optimization (PSO) [10], Gravitational Search Algorithm (GSA) [27], Moth-Flame Optimization (MFO) [20], Genetic algorithm (GA) [17]. The bonobo optimizer (BO) is a novel meta-heuristic algorithm that is modeled on bonobo social behaviour [6]. The BO's search capabilities was made more resilient and efficient by using a fission-fusion approach for selection, four well-developed methods for producing offspring, and the application of two distinctive phases.

Despite the fact that population based approaches can grant promising solutions for optimization problems, as the dimension of the search space grows, they experience a series of challenges. A major key challenges is that, population based approaches frequently become stuck in local optimum when dealing with multi-modal complicated problems [3]. Correspondingly, to achieve high performance on complicated optimization problems, the exploration and exploitation stages should be well balanced [12].

The photovoltaic (PV) solar model presents one of the most exciting themes that has led to an increase in researchers' interest [13, 28]. However, one of the key challenges for researchers is to insure that the PV model captures the maximum amount of available power [4]. Different PV solar models have been presented including the static and dynamic PV models. Nevertheless, in the static PV model the representation of the load connection, switching and variation is not taken into account. Therefore, dynamic PV model has been proposed to overcome the drawbacks of the static PV models by representing the load connection in the PV model [8, 16]. The accuracy of dynamic PV models is essentially impacted by the precision of their parameters values which are obtained under various operating conditions. Accordingly, an accurate identification of the PV parameters is vital to gain the maximum power of the dynamic PV model.

Several conventional methods are applied for the PV parameters identification [14, 29]. Meanwhile, for dynamic PV models, only the non-linear least square approaches and least square have been utilized for parameters identification [1, 8]. Nevertheless, conventional methods based on classical numerical/analytical tools might be unable to accurately fit with the PV model, in consequence of the multi-modal and non-linear nature of the problem, which leads to negatively impact the maximum available power optimal capturing.

Motivated from the above discussion, a quantum behaved bonobo optimization algorithm (QBOA) is proposed. Whereby, quantum mechanics are adopted in this paper to integrate a quantum behavior in the bonobo optimization algorithm. For which, the proposed quantum-behaved bonobo optimization exploitation mechanism

absorbs the character of Quantum-behaved method. Aiming to evaluate the robustness and coherence of the QBOA, the proposed QBOA performance is evaluated on CEC2019 and CEC2005 benchmark. Furthermore, the QBOA algorithm was adapted to optimize the dynamic photovoltaic models parameters.

The major contributions of this paper are as follows:

1. A quantum behaved bonobo optimization algorithm is proposed (QBOA), which is combining the advantages of the BOA and quantum mechanics to direct the search agents through the search space.
2. Several tests are conducted over unimodal and multimodal benchmark functions that are adopted for assessing the effectiveness of the proposed QBOA algorithm.
3. The proposed QBOA algorithm is used for optimizing integral and fractional dynamic photovoltaic models. The experimental results ensure that the QBOA algorithm is efficient enough in the identification of the dynamic PV model parameters.

The remainder of the paper is organized as follows: In Section 2, the Bonobo Optimization algorithm brief description is given. In Section 3, the proposed QBOA is detailed. In Section 5, the efficiency of the proposed QBOA algorithm on CEC2019 and CEC2005 benchmarks, as well as comparative analysis of the QBOA versus several optimization algorithms are presented. The dynamic photovoltaic models are provided in Section 4. The simulation results and analysis of the QBOA of dynamic PV models are discussed in Section 5.4. Lastly, in Section 6, The paper's key findings are discussed.

2. Bonobo optimization algorithm

Das and Pratihar presented the Bonobo Optimizer (BO) as a new metaheuristic algorithm [6]. The BO algorithm simulates the reproductive approaches social behavior of bonobos. BO mimics the fission-fusion process, which focuses on segmenting the community into multiple subgroups of varying compositions and sizes, then rejoining them with the rest of the community. Restricting, promiscuous, consortship and extra-group mating are the four types of bonobo strategies. The BO algorithm's working principle is depicted in detail as follows:

Non-user initial parameters are set: the positive and negative phase count $ppc = 0, npc = 0$, the change in phase $cp = 0$, the extra-group mating probability $p_{xgm} = p_{xgm-initial}$, the sizing factor of the temporary sub-group $tsgs_{factor} = tsgs_{factor-initial}$, the directional probability $p_d = 0.5$ and the phase probability $p_p = 0.5$.

Inspired by the fission-fusion social group technique [7], the p^{th} bonobo is chosen for pairing with the i^{th} bonobo. For which, the temporary subgroup maximum size $tsgs_{max}$ is calculated using the following equation:

$$tsgs_{max} = \max(2, tsgs_{factor} \cdot N) \quad (1)$$

Where N is the size of the population. According to Equation (1), the temporary subgroup size is between 2 and $tsgs_{max}$; and is constructed by randomly selecting non-repeats bonobos from the $N - 1$ population, eliminating the i^{th} bonobo. The optimal solution from the subgroup is picked as the p^{th} bonobo if its fitness is higher than the i^{th} bonobo; contrarily, a randomly bonobo from the subgroup is selected as the p^{th} bonobo. The chosen p^{th} bonobo then begins mating in order to generate the offspring.

In the bonobo society, four different types of mating have been observed: extra-group mating, promiscuous, consortship and restrictive mating. The mating method differs according to whether the phase is positive or negative. The possibility of restricted and promiscuous matings is high during a positive phase. On the other hand, in a negative phase, extra-group and consortship matings are perceived as high. A mating method is utilized using the phase probability parameter p_p . Whereby, a random number $r \in [1, 0]$ is generated, and determined to be either equal or less than p_p . A new _bonobo is created using the following equation:

$$\begin{aligned} new_bonobo_k = & bonobo_k^i + r_1 \cdot scab \cdot (\alpha_k^{bonobo} - bonobo_k^i) + \\ & (1 - r_1) \cdot flag \cdot scsb \cdot (bonobo_k^i - bonobo_k^p) \end{aligned} \quad (2)$$

where r_1 is a random number in $[0, 1]$ and k is the optimization problem decision variable number. While, $scsb$ and $scab$ are the p^{th} bonobo and the alpha bonobo sharing coefficients, respectively. The parameters $scab$ and $scsb$ are predetermined constants that affect the balance between explorative and exploitative tendencies. The flag parameter can have two possible values: 1 or -1 for promiscuous mating or restrictive mating, respectively.

Sharing coefficients for the alpha-bonobo and p^{th} -bonobo are represented using $scab$ and $scsb$, respectively.

On the contrary, extra-group mating is employed to create an offspring when r is bigger than p_p , as shown below:

$$\beta_1 = e^{(r_4 + r_4^2 - 2/r_4)} \quad (3)$$

$$\beta_2 = e^{(2r_4 - r_4^2 - 2/r_4)} \quad (4)$$

$$new_bonobo_k = bonobo_k^i + \beta_1 \cdot (UB_k - bonobo_k^i) \quad (5)$$

$$new_bonobo_k = bonobo_k^i - \beta_2 \cdot (bonobo_k^i - LB_k) \quad (6)$$

$$new_bonobo_k = bonobo_k^i - \beta_1 \cdot (bonobo_k^i - LB_k) \quad (7)$$

$$new_bonobo_k = bonobo_k^i + \beta_2 \cdot (UB_k - bonobo_k^i) \quad (8)$$

where, LB_k and UB_k are the lower and upperboundary, respectively; while, $r_4 \neq 0$ is a random number.

If a random number r_2 is greater than p_{xgm} , the consorship mating method is utilized to generate an offspring, as follows:

$$new_bonobo_k = \begin{cases} bonobo_k^i + flag \cdot e^{-r_5} \cdot (bonobo_k^i - bonobo_k^p), & \text{if } (r_6 \leq p_d || flag = 1) \\ bonobo_k^p, & \text{otherwise} \end{cases} \quad (9)$$

When the new bonobo's fitness is discovered to be better than the parent's, or when a random number $r \in [0, 1]$ is equal to or less than p_{xgm} , the new bonobo is accepted. Furthermore, if the new_bonobo fitness is shown to be superior to the alpha fitness, the new_bonobo is designated as the alpha-bonobo.

The BO's controlling parameters are modified as follows when the newly obtained alpha bonobo in the current iteration is detected to be an improved solution.

$$npc = 0, cp = \min(0.5, ppc \cdot rcpp), ppc = ppc + 1$$

$$p_{xgm} = p_{xgm-initial}, p_d = p_p, p_p = cp + 0.5$$

$$tsgs_{factor} = \min(tsgs_{factor-max}, (tsgs_{factor-initial} + ppc \cdot rcpp^2))$$

Where $rcpp$ is the change rate of the phase probability. On the other hand, the following updates have been made to the controlling parameters:

$$npc = 1 + npc, ppc = 0, cp = -\min(0.5, rcpp \cdot npc)$$

$$p_p = cp + 0.5, p_{xgm} = \min(0.5, (p_{xgm-initial} - rcpp^2 \cdot npc))$$

$$tsgs_{factor} = \min(0, (tsgs_{factor-initial} - npc \cdot rcpp^2)), p_d = p_p$$

3. Proposed quantum-behaved Bonobo optimization algorithm (QBOA)

In Bonobo optimizer, the bonobos are characterized by their location and position vector, which constitute the bonobo particle's trajectory. In accordance with Newtonian mechanism particles moves along a predetermined trajectory. However, due to the principle of uncertainty, it is not possible to estimate both distance and position simultaneously in reality.

Accordingly, quantum mechanics are adopted in this study to integrate quantum behaviour in the bonobo optimization algorithm. For which, the proposed quantum-behaved bonobo optimization exploitation mechanism absorbs the character

of Quantum-behaved method. The mechanism setting places the bonobo in quantum mechanics space, utilizes the wave function to describe the bonobo's position and regulated the bonobo's state change process according to the Schrodinger equation [18].

In quantum mechanics, the fundamental time dependent Schrodinger equation is defined by:

$$i\hbar \frac{\partial \Psi}{\partial t} = -\hat{H}(X)\Psi \quad (10)$$

The wave function Ψ described the quantum state of the bonobo and only depends on its position. $\hat{H}(X)$ is a time independent Hamiltonian operator given by:

$$\hat{H}(X) = -\frac{\hbar^2}{2m} \nabla^2 + V(X) \quad (11)$$

Where \hbar is the Planck's constant, $V(X)$ is the potential energy distribution and m is the bonobo mass. In a three dimensional space, the probability density of the bonobo in a position to appear is given by:

$$|\Psi|^2 dx dy dz = Q dx dy dz \quad (12)$$

Where, Q is the probability density function that meets the normalization condition:

$$\int_{-\infty}^{+\infty} |\Psi|^2 dx dy dz = \int_{-\infty}^{+\infty} Q dx dy dz = 1 \quad (13)$$

Moreover, the positions of local attractor for each bonobo can be defined as:

$$b_k = scab \cdot r \cdot \alpha_k^{bonobo} + scsb \cdot (1 - r) \cdot bonobo_k^p \quad (14)$$

The statistical justification for the wave function is shown by Equations (12) and (13) where the integration is carried out over the full space. Each bonobo in the proposed algorithm has assumed a spin-less movement with a specific potential energy in D-dimensional Hilbert space. The bonobo is pulled using this field in accordance with a position specified by Equation (14).

For D-dimensional Hilbert space, where each bonobo position is bounded by delta potential well; a new_bonobo is created using the following equation:

$$new_bonobo_k = b_k + flag \cdot \alpha_k^{bonobo} \cdot |bonobo_k^i - W_k| \cdot \ln\left(\frac{1}{r}\right) \quad (15)$$

Where r is a random number in $[0,1]$ and W is the average positions of the bonobo at iteration t . The pseudo code of the QBOA algorithm is presented in Algorithm 1.

Algorithm 1 Pseudocode of the QBOA**Input:**Total population number N_{pop} Optimization iterations number Max_Iter **Output:**

Optimal alpha Bonobo

```

1: Initialize the bonobo parameters
2: Initialize the bonobo population positions randomly.
3: Calculate the objective values for each search agent and dictate the  $\alpha$  bonobo
4: while  $t \leq Max\_Iter$  do
5:   Calculate  $tsgs_{max}$  using equation 1
6:   for  $i=1:N_{pop}$  do
7:     Determine the temporary subgroup size
8:     #Apply the fission-fusion Technique
9:     Select Flag value
10:    if  $r \leq p_p$  then
11:      #Update the position of the bonobo using the quantum mechanism
12:      Calculate the positions of local attractor for each bonobo using equation 14
13:      Create new_bonobo using equation 15
14:      Apply the boundary limiting conditions
15:    else
16:      for  $k=1:d$  do
17:        if  $r_2 < p_{xgm}$  then
18:          if  $\alpha^{bonobo}_k \geq bonobo_k$  then
19:            if  $r_3 \leq p_p$  then
20:              Create new_bonobok using equation 5
21:            else
22:              Create new_bonobok using equation 6
23:            end if
24:          else
25:            if  $r_4 \leq p_p$  then
26:              Create new_bonobok using equation 7
27:            else
28:              Create new_bonobok using equation 8
29:            end if
30:          end if
31:        else
32:          Create new_bonobok using equation 9
33:        end if
34:      Apply the boundary limiting conditions
35:    end for
36:    Evaluate the new_bonobok fitness value
37:    if  $fitness(new\_bonobo_k) < fitness(\alpha^{bonobo})$  then
38:       $\alpha^{bonobo} = new\_bonobo_k$ 
39:    end if
40:  end if
41: end for
42: Update the controlling parameters
43:  $t=t+1$ 
44: end while
45: return  $\alpha^{bonobo}$ 

```

4. Dynamic photovoltaic models

4.1. Integral dynamic photovoltaic model

The considered integral dynamic PV model [8] is a second-order model that account the junction capacitance and conductance, along with the inductive effects, as shown in Figure 1a. The PV model and its associated load are valid in the area of the current-voltage curve which lies between the near constant voltage region and the open circuit voltage [9,15]. The circuit in Figure 1a has a linear behavior; accordingly, it is possible to reduce the PV static part to a series resistance R_s and a constant voltage source V_{oc} , yielding the circuit in Figure 1b. While, the dynamic part of the integral PV model is represented by conductance R_c , capacitor C for junction capacitance, and inductance L for cabling and connection inductance.

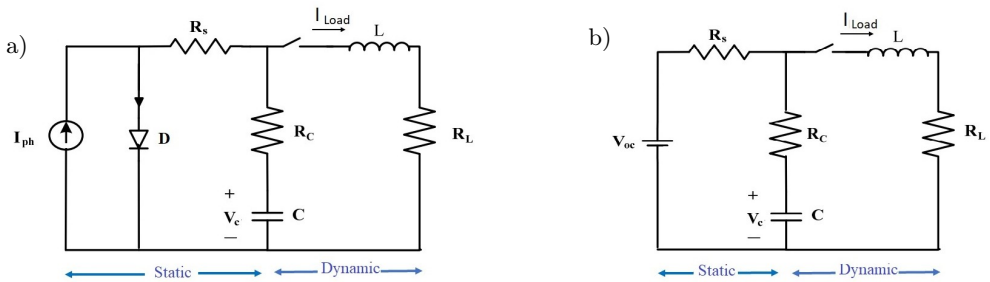


Figure 1. Integral-dynamic PV model: PV model (a); equivalent PV model in the linear voltage region (b)

4.2. Fractional dynamic photovoltaic model

The inductor and capacitor in the fractional dynamic PV model are alternated with fractional equivalents of orders β and α , respectively, as illustrated in Figure 2.

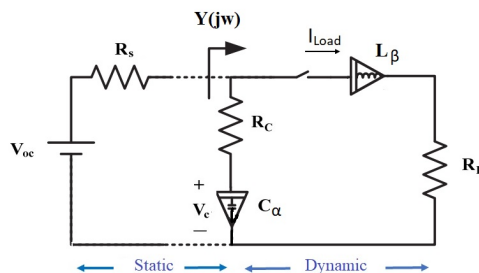


Figure 2. Fractional-dynamic PV model

Whereby, the fractional capacitor's effect is visible in low value of the resistor R_c ; due to, the real frequency dependence on fractional capacitance impedance [26].

5. Simulation results and analysis

With the goal of examining the performance and capabilities of the proposed Quantum-Behaved Bonobo Optimization Algorithm QBOA, Matlab R2018b was adjusted for simulation purposes. All evaluation experiments were conducted on: Intel(R), Core i7 4910MQ CPU@2.90GHz and 16GB RAM.

5.1. Performance estimation with CEC 2005

With the aim to evaluate the proposed QBOA algorithm performance, multiple optimization test problems solved over various runs to obtain a reliable conclusion. QBOA is estimated on 23 test functions extracted from the CEC 2005 [19]. Accordingly, the test functions are separated into two categories based on their characteristics: ($F1 - F7$) unimodal and ($F8 - F23$) multimodal functions. Whereby, the test objective functions are denoted as: "differentiable, non-differentiable, discontinuous, continuous, scalable, non-scalable, non-separable and separable". Since they include no local optima and a single global optimum, unimodal functions are used to estimate the exploitative potential of the meta-heuristic algorithm. Multimodal functions, on the other hand, have several local optimal and one global optimum. As a result, they can be used to assess the meta-heuristic algorithm's capability for exploration and escape from local optima.

The BOA and proposed QBOA internal parameters are adapted as: total population number $N = 50$, stopping criterion of $30000 \cdot d$, where d is the optimization problems dimension, $p_{xgm-initial} = 1/d$, $rcpp = 0.0036$, $tsgs_{factor-max} = 0.02$, and the sharing coefficients $scab = 1.3$ and $scsb = 1.4$. To conduct an unbiased comparison, the statistical results for each test function are calculated across 30 independent runs with completely random initial conditions. Through which, four distinct evaluation factors are taken into account: the minimum (best) solution, the average (mean) solution, the maximum (worst) solution and the standard deviation (St.dev). The worst, best and mean metrics examine the accuracy of the solution, while the St.dev estimates the obtained solution's robustness.

The experimental finding of the BOA and proposed QBOA; on the unimodal and multimodal test optimization functions are recorded in Table 1. From Table 1, it is clear that the proposed quantum-behaved BO surpass the BO algorithm in term of mean, best and worst results, except for test function F12. Both QBOA and BOA could continuously attain the global optimal for F5 and F6. Furthermore, compared to the BOA, the QBOA was able to locate the global optima with less standard deviation for all test functions, which demonstrate the QBOA's robustness in locating the global optimal.

Table 1
Statistical results of BOA and QBOA algorithm on CEC 2005

Function	BOA				QBOA			
	Best	Mean	Worst	St.dev	Best	Mean	Worst	St.dev
F1	9.5176E-57	4.51667E-46	1.35358E-44	2.4712E-45	1.3325E-141	2.0989E-118	5.3473E-117	9.7943E-118
F2	9.18109E-29	3.80198E-26	2.25682E-25	6.34863E-26	5.29453E-72	3.15854E-60	9.45349E-59	1.72583E-59
F3	3.01466E-56	5.08291E-50	6.43562E-49	1.33984E-49	9.6229E-136	1.8777E-109	5.633E-108	1.0284E-108
F4	4.27144E-28	3.31868E-25	2.49976E-24	6.22758E-25	9.13827E-72	6.93219E-57	2.05336E-55	3.74755E-56
F5	0	0	0	0	0	0	0	0
F6	0	0	0	0	0	0	0	0
F7	1.17E-04	1.49E-03	4.23E-03	1.09E-03	2.84E-06	7.58E-04	2.34E-03	5.87E-04
F8	-418.9828873	-411.0869983	-300.5445527	30.0487686	-4189.982887	-4182.982887	-4111.982869	3.31728E-06
F9	0	0.033165302	0.994959057	0.18165384	0	0	0	0
F10	8.88178E-16	8.88178E-16	8.88178E-16	0	8.88178E-16	8.88178E-16	8.88178E-16	0
F11	0	0.001644112	0.009864672	0.003739194	0	0	0	0
F12	4.71163E-31	4.71163E-31	4.71163E-31	8.90784E-47	7.81257E-26	3.69146E-08	1.10744E-06	2.0219E-07
F13	4.69E-27	3.0545E-06	6.62661E-05	1.28046E-05	1.34978E-32	1.34978E-32	1.34978E-32	5.5674E-48
F14	0.998003838	1.776170636	12.67050581	2.961408668	0.998003838	0.998003838	0.998003838	1.00999E-16
F15	3.08E-04	4.58E-03	2.04E-02	7.51E-03	3.09E-04	3.30E-03	2.04E-02	6.81E-03
F16	-1.031628453	-1.031628453	-1.031628453	5.21556E-16	-1.03	-1.03	-1.03	0
F17	0.397887358	0.397887358	0.397887358	0	0.397887358	0.39788736	0.397887	0
F18	3	3.9	30	4.929503018	3	3	3	1.30E-15
F19	-3.862779787	-3.837012599	-3.089764134	0.141132703	-3.86	-3.86	-3.86	7.05E-15
F20	-3.042457738	-3.017878829	-2.981002427	0.030617435	-3.042423011	-3.025796432	-2.975483131	0.026197581
F21	-10.15319876	-9.984784532	-5.100772055	0.922442691	-10.15319876	-10.1531982	-10.15318981	2.12097E-16
F22	-10.40282204	-10.05122207	-5.128822495	1.338056572	-10.40282204	-10.40232042	-10.39505238	0.001910066
F23	-10.5362903	-9.998653738	-5.128480623	1.640498374	-10.5362903	-10.53629027	-10.53628927	1.87452E-17

The proposed QBOA was analyzed versus six well known met-heuristic algorithms: Coyote Optimization Algorithm (COA) [24], Dragonfly Algorithm (DA) [21], Particle Swarm Optimization (PSO) [10], Gravitational Search Algorithm (GSA) [27], Moth-Flame Optimization (MFO) [20], Genetic algorithm (GA) [17], as shown in Table 2. The initial controlling parameters of the optimization algorithms are listed in Table 3.

Table 2
Comparison results attained for QBOA and different optimization algorithms

Function		QBOA	COA	PSO	DA	GSA	GA	MFO
F1	Mean	2.0989E-118	25.32456	1.36E-04	5.30E-01	2.53E-16	8.00E-04	1.65E-31
	St.dev	9.7943E-118	9.284834	2.02-7	1.318	9.67E-17	8.70E-04	4.91E-31
F2	Mean	3.15854E-60	0.7051718	0.042144	2.392	0.055655	3.00E-03	2.69E-19
	St.dev	1.72583E-59	0.1208514	0.045421	3.912	0.194074	1.80E-03	6.22E-19
F3	Mean	1.8777E-109	2252.63547	70.12562	215.45	896.5347	13.213	2.05E-11
	St.dev	1.0284E-108	825.3839	22.11924	935.17	318.9559	8.042	4.21E-11
F4	Mean	6.93219E-57	24.4519057	1.086481	1.153	7.35487	0.209	5.79E-06
	St.dev	3.74755E-56	3.6646211	0.317039	2.702	1.741452	5.80E-02	3.17E-05
F5	Mean	0	2592.44628	96.71832	6784.5	67.54309	66.9	133.11
	St.dev	0	1925.75963	60.11559	21974.5	62.22534	22.6	555.57
F6	Mean	0	27.835087	0.000102	2.2023	2.50E-16	7.50E-04	4.78E-32
	St.dev	0	12.9163617	8.28E-05	5.528	1.74E-16	7.20E-04	1.27E-31
F7	Mean	7.58E-04	6.72E-02	1.23E-01	6.90E-03	0.089441	8.10E-04	1.20E-03
	St.dev	5.87E-04	2.40E-02	4.50E-02	7.60E-03	0.04339	5.50E-04	7.20E-04
F8	Mean	-4182.982887	-12299.311	-4841.29	-3213.66	-2821.07	-3692.39	-3329.13
	St.dev	3.31728E-06	105.679038	1152.814	431.748	493.0375	182.42	288.317
F9	Mean	0	20.11984	46.70423	11.561	25.96841	3.80E-04	12.8372
	St.dev	0	2.22484365	11.62938	10.177	7.470068	3.20E-04	7.352
F10	Mean	8.88E-16	4.0391476	2.76E-01	3.14E-05	6.21E-02	8.88E-16	8.88E-16
	St.dev	0	0.95061505	5.09E-01	1.70E-04	2.36E-01	1.00E-31	1.00E-31
F11	Mean	0	1.1976333	9.22E-03	0.3846	27.70154	8.88E-16	1.78E-01
	St.dev	0	7.64E-02	7.72E-03	0.3826	5.040343	1.00E-31	8.43E-02
F12	Mean	3.69146E-08	2.0935602	6.92E-03	0.5296	1.799617	5.73E-05	3.11E-02
	St.dev	2.0219E-07	0.809141215	2.63E-02	0.6912	0.95114	1.40E-04	9.49E-02
F13	Mean	1.34978E-32	11.91776	6.68E-03	0.5292	8.899084	6.21E-05	1.10E-03
	St.dev	5.5674E-48	3.7510395	8.91E-03	0.7173	7.126241	1.10E-04	3.33E-03
F14	Mean	0.998	0.998	3.627168	1.1	3.4	0.998	1.03
	St.dev	1.00999E-16	4.12E-17	2.560828	0.303306	2.578637	8.83E-14	0.181483682
F15	Mean	3.30E-03	3.07E-04	5.77E-04	1.34E-03	1.80E-03	8.40E-04	8.37E-04
	St.dev	6.81E-03	7.70E-09	2.22E-04	5.11E-04	4.90E-04	2.90E-04	2.54E-04
F16	Mean	-1.03	-1.03	-1.03	-1.03	-1.03	-1.03	-1.03
	St.dev	0	6.58E-16	6.25E-16	2.55E-11	0	5.02E-10	0
F17	Mean	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
	St.dev	0	0	0	7.60E-13	0	4.73E-07	1.13E-16
F18	Mean	3	3	3	3	3	3	3
	St.dev	1.30E-15	1.36E-15	1.33E-15	1.38E-06	4.17E-15	1.21E-08	1.95E-15
F19	Mean	-3.86	-3.86277	-3.8628	-3.86	-3.8628	-3.86	-3.86
	St.dev	7.05E-15	3.12E-15	2.58E-15	1.59E-03	2.29E-15	2.20E-03	2.71E-15

Table 2 cont.

Function		QBOA	COA	PSO	DA	GSA	GA	MFO
F20	Mean	-3.025796432	-3.04245773	-3.26634	-3.25	-3.31778	-3.32	-3.22
	St.dev	2.62E-02	2.21E-13	6.05E-02	6.72E-02	2.31E-02	2.17E-02	4.51E-02
F21	Mean	-10.1531982	-10.153199	-9.31	-9.81	-9.95512	-10.2	-7.56
	St.dev	2.12097E-16	7.23E-15	1.925505	1.280913	3.737079	4.84E-04	3.323037
F22	Mean	-10.40232042	-10.40232042	-9.52	-10.4	-9.68447	-9.93	-9.35
	St.dev	1.91E-03	0.962917757	2.00228	0.192434	2.014088	1.822252	2.423664
F23	Mean	-10.536	-10.536	-10.536	-10.536	-10.536	-9.61	-10.3
	St.dev	1.87452E-17	1.22342651	1.635722	1.060781	2.60E-15	2.405191	1.39948

Table 3

Parameter settings of the optimization algorithms

Optimization Algorithm	Parameter Setting
Quantum-Behaved Bonobo Optimization Algorithm (QBOA)	rcpp = 0.0036, $tsgs_{factor-max} = 0.02$, scab = 1.25, scsb = 1.3
Particle Swarm Optimization (PSO)	$c1 = c2 = 2$, $w_{max} = 0.9$, $w_{min} = 0.2$
Coyote Optimization Algorithm (COA)	packs number $N_p = 10$, coyotes number $N_c = 10$
Gravitational Search Algorithm (GSA)	$G_0 = 100$, $alpha = 20$
Genetic Algorithm (GA)	Roulette wheel selection, crossover = 0.7, mutation = 0.3
Dragonfly Algorithm (DA)	$\beta = 0.5$
Moth-Flame Optimization (MFO)	b = 1, a decreased linearly from -1 to -2

As reported in Table 2, the proposed QBOA algorithm surpassed the comparison algorithms for all optimization test functions except for F8, where COA presents better mean and QBOA presents the second best; and for F15 where COA presents better standard deviation and mean measure. Compared to the GA and MFO algorithm, QBOA finds the second best results for function F20. Moreover, for functions F5, F6, F9, and F11, the theoretical global optimum could be consistently located through QBOA. While, for test function F10 QBOA was able to attain the theoretical global optimal similar to GA and MFO algorithms, however with the best St.dev.

5.2. Performance estimation with CEC 2019

Extra examinations of the proposed quantum behaved BO on the CEC2019 benchmarks are undertaken in this sub-section. CEC2019 [25] signifies a test environment, including ten different functions with various characteristics. CEC01, CEC02, and CEC03 are the first three functions, with dimensions of: $[-8192, 8192]$, $[-16384, 16384]$ and $[-4, 4]$, respectively. While, the test functions CEC04 to CEC10 are shifted and rotated in the range $[-100, 100]$. The evaluation findings are reported in Table 4. From Table 4, the proposed QBOA algorithm provides the best results in term of best, worst, mean and standard deviation for all test functions.

Table 4
Statistical results of BOA and QBOA algorithm on CEC 2019

Function	BOA					QBOA				
	Best	Mean	Worst	St.dev	Best	Mean	Worst	St.dev		
CEC01	5.05E+09	4.39E+10	8.01646E+11	1.45922E+11	9.44E+07	3.82E+10	1.67726E+11	35343962073		
CEC02	17.54313222	29.24788963	147.0449426	27.2284062	17.34285715	17.34394919	17.37353053	0.000158971		
CEC03	12.70240428	12.70240486	12.70241987	2.88462E-06	12.70240422	12.70240826	12.70242195	1.13E-07		
CEC04	196.933194	1224.912295	4116.441557	835.165023	32.78727204	188.1486484	1342.768471	24.1737793		
CEC05	1.462741151	1.963403865	2.282512869	0.18092835	1.041741434	1.105880945	1.935413404	0.021160053		
CEC06	8.32434066	11.14097169	13.38913543	1.333579119	8.028343674	9.275948809	13.29664279	1.024430079		
CEC07	230.2766639	841.5797016	1358.830208	290.1988289	105.5116356	410.001414	1263.376387	186.1911856		
CEC08	5.80548777	6.799254532	7.18450629	0.990889863	3.71901424	5.099504448	7.585492774	0.401191137		
CEC09	5.030062803	53.47166279	205.4756953	52.53316654	2.075989319	2.163408587	5.849287551	0.008198		
CEC10	20.2806579	20.57014903	20.78783798	0.135932697	20.26194419	20.13350873	20.77914636	0.115771177		

In addition, the QBOA is examined against four well known algorithms in the literature: Particle Swarm Optimization (PSO) [10], Dragonfly Algorithm (DA) [21], Whale optimization algorithm (WOA) [23] and Salp swarm algorithm (SSA) [22], Table 5. From Table 5, the proposed QBOA Obtains better results for CEC02, CEC05, CEC06, CEC08 and CEC09. Furthermore, QBOA provides the better mean for CEC10, and produces the second-best outcomes for test function CEC07 in comparison to the PSO algorithm.

Table 5

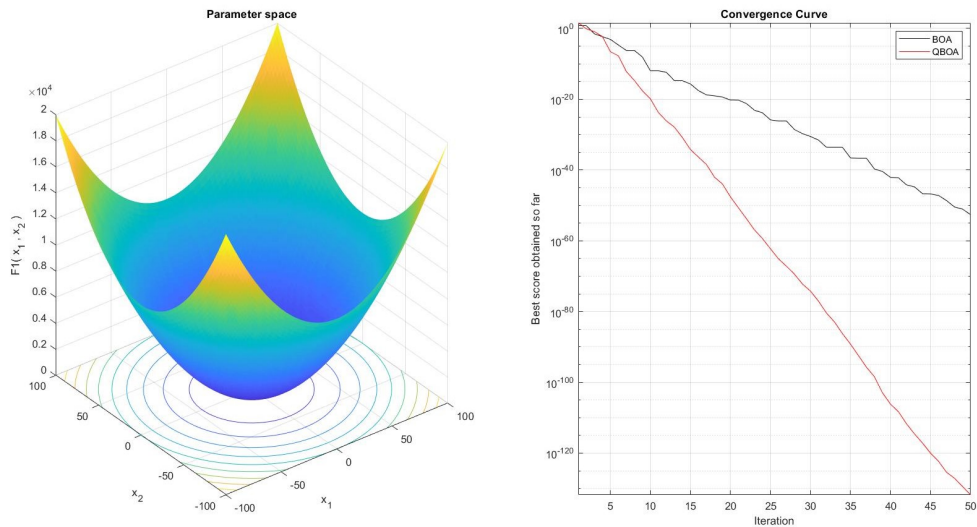
Comparison results attained for QBOA and different optimization algorithms on CEC2019

Function		QBOA	PSO	DA	WOA	SSA
CEC01	Mean	3.82E+10	1.471E+12	5.43E+10	4.11E+10	6.05E+09
	St.dev	3.53E+10	1.324E+12	6.69E+10	5.42E+10	4.75E+09
CEC02	Mean	17.34394919	15183.91348	78.0368	17.3495	18.3434
	St.dev	0.000158971	3729.553229	87.7888	0.0045	0.0005
CEC03	Mean	12.7024	12.7024	13.7026	13.7024	13.7025
	St.dev	1.13E-07	9.03E-15	0.0007	0	0.0003
CEC04	Mean	188.1486484	16.80077558	344.3561	394.6754	41.6936
	St.dev	24.1737793	8.199	414.0982	248.5627	22.2191
CEC05	Mean	1.105880945	1.138264	2.5572	2.7342	2.2084
	St.dev	0.021160053	0.08938	0.3245	0.2917	0.1064
CEC06	Mean	9.275948809	9.30531	9.8955	10.7085	6.0798
	St.dev	1.024430079	1.69	1.6404	1.0325	1.4873
CEC07	Mean	410.001414	160.686	578.9531	490.6843	410.3964
	St.dev	186.1911856	104.203	329.3983	194.8318	290.5562
CEC08	Mean	5.099504448	5.2241	6.8734	6.909	6.3723
	St.dev	0.401191137	0.7867	0.5015	0.4269	0.5862
CEC09	Mean	2.163408587	2.373279	6.0467	5.9371	3.6704
	St.dev	0.008198	0.01843	2.871	1.6566	0.2362
CEC10	Mean	20.13350873	20.2806	21.2604	21.2761	21.04
	St.dev	0.115771177	0.12853	0.1715	0.1111	0.078

5.3. Convergence analysis

The proposed QBOA and BOA's convergence behaviour are explored. The cost function for F1–F4, F7, F9, F10, F13, F15, and F21 test problems, as well as their associated convergence curves, are shown in Figures 3–6. As shown in Figure 3, the QBOA algorithm exhibits abrupt changes in the early stages of iterations, which decreased gradually throughout the course of iterations. This behaviour demonstrates that, the proposed QBOA algorithm gains from a well balance of exploitation and exploration, accordingly enables the QBOA to avoid being locked into local optimals.

a)



b)

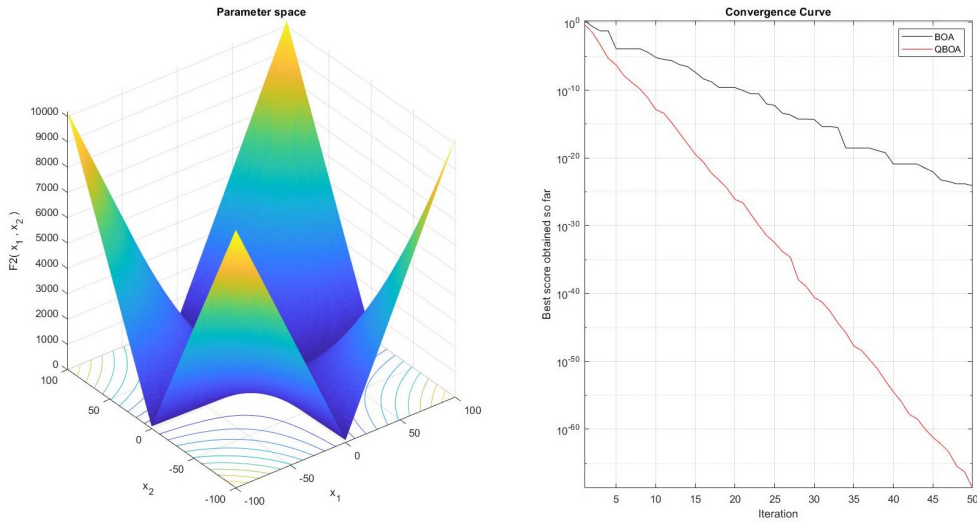
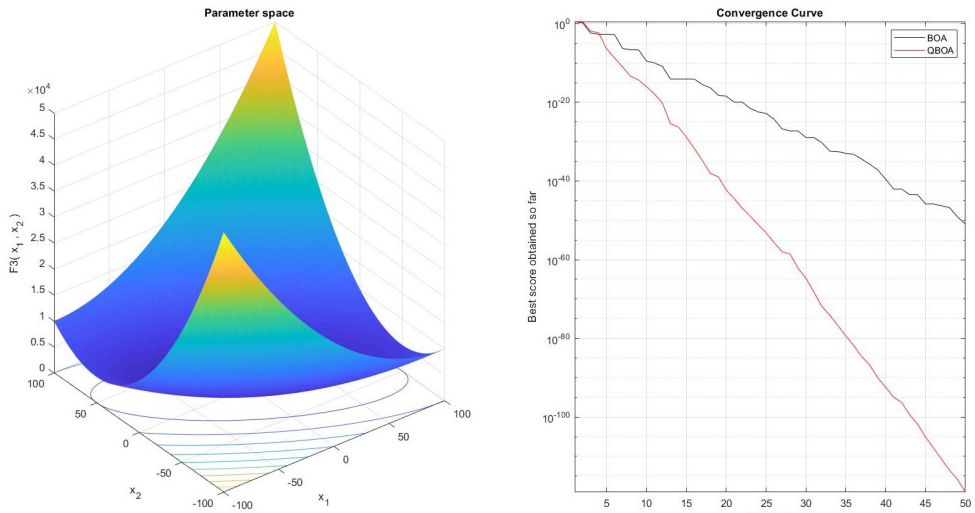


Figure 3. Best fitness convergence curves: a) $F1$; b) $F2$

a)



b)

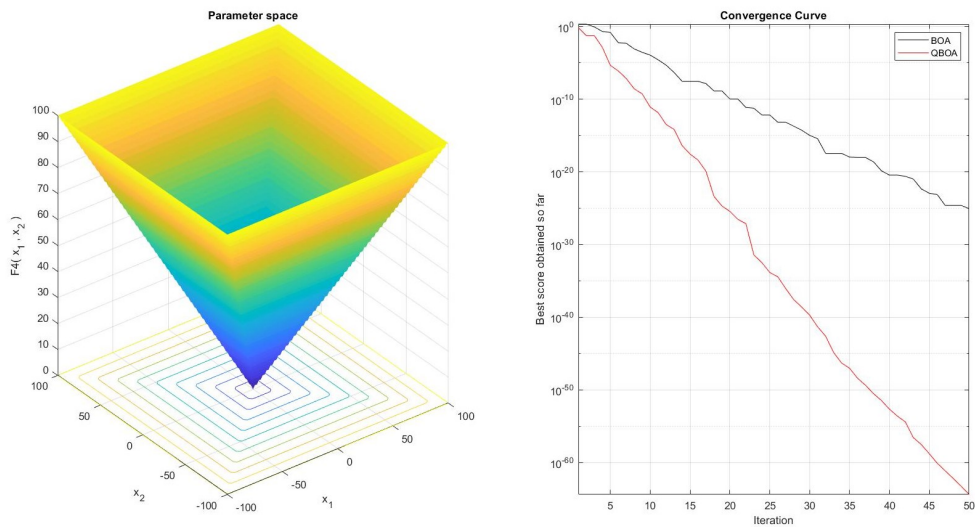
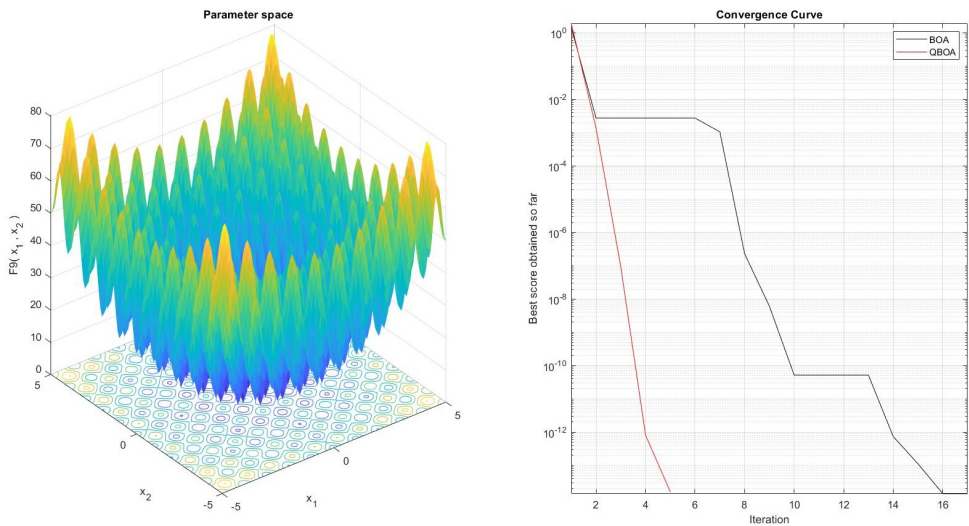


Figure 4. Best fitness convergence curves (cont.): a) $F3$; b) $F4$

a)



b)

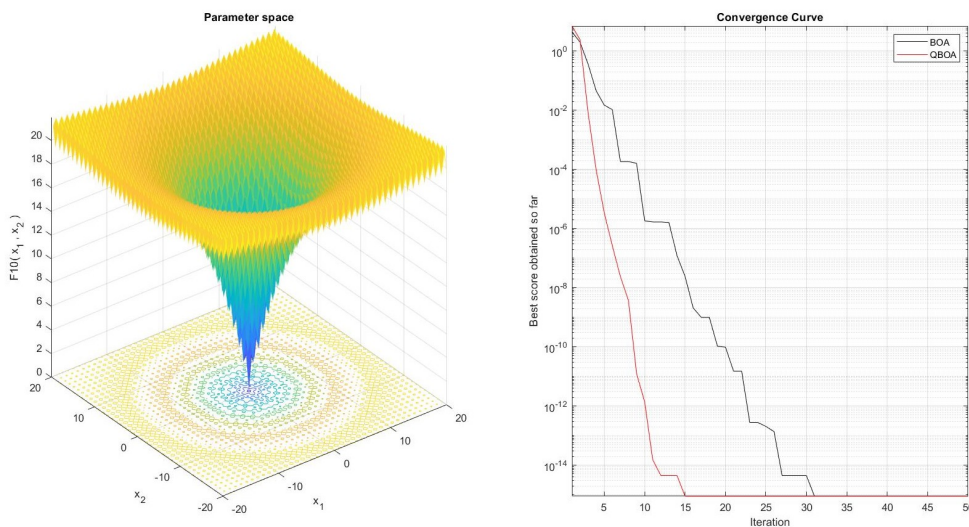
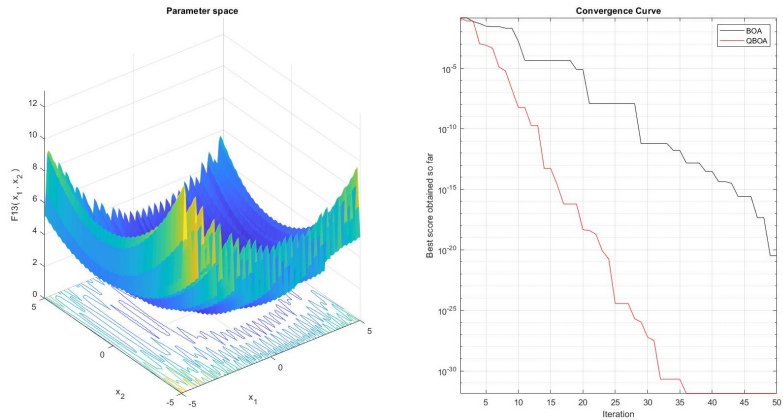
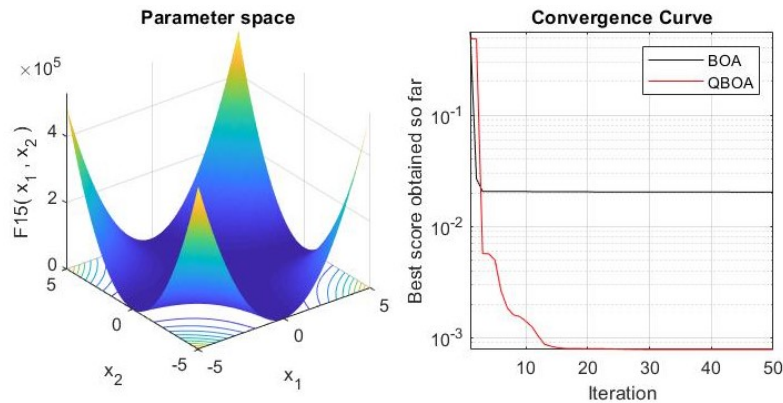


Figure 5. Best fitness convergence curves (cont.): a) $F9$; b) $F10$

a)



b)



c)

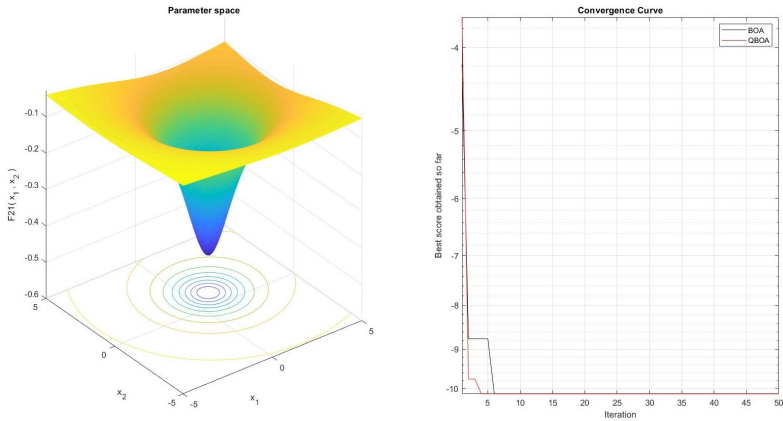


Figure 6. Best fitness convergence curves (cont.): a) $F13$; b) $F15$; c) $F21$

5.4. Dynamic PV models' parameters estimation

This subsection is concerned with the application of the proposed QBOA for parameter optimization of the integral and fractional dynamic PV models. The dynamic data sets were collected from PV module at an irradiance level of 655 W/m^2 and a temperature of 25°C , with connected load $R_L = 23.1 \Omega$ [8]. Whereby, for the integral dynamic PV model, three unknown parameters R_c, L and C should be estimated, while in the fractional dynamic PV model five parameters $R_c, L_\beta, C_\alpha, \beta$ and α should be estimated.

Thirty independent runs of the PV optimization problems were performed. The BOA and proposed QBOA internal parameter values for optimizing the dynamic PV models were kept the same as follows: $N = 50$, $p_{xgm-initial} = 1/d$, $rcpp = 0.0036$, $tsgs_{factor-max} = 0.02$, $scab = 1.3$ and $scsb = 1.4$. The upper and lower boundaries of the fractional and integral dynamic PV models are given in Table 6.

Table 6
Dynamic PV model parameters boundaries

Model	Parameters	Lower bound (LB)	Upper bound (UB)
Integral PV	R_c	0	20
	C	2.0E-08	600 E-07
	L	5.0 E-06	100 E-06
Fractional PV	R_c	0	20
	C_α	2.0E-08	600 E-07
	L_β	5.0 E-06	100 E-06
	α	0.8	1.1
	β	0.8	1.1

The three optimized parameters (R_c, L, C) for the integral PV model, the five optimized parameters ($R_c, C_\alpha, \alpha, L_\beta, \beta$) for the fractional PV model and their corresponding objective function (RMSE) are reported in Table 7 and Table 8. Moreover, the proposed QBOA's RMSE findings are compared to BOA algorithm and various well-known and recently developed optimizers, including Gradient-Based Optimizer (GBO) [2], artificial ecosystem based optimization (AEO) [30] and jellyfish search optimizer (JS) [5]. The tabulated results revealed that, the proposed QBOA produced the best RMSE values.

For more comprehensive validation of the proposed QBOA, the load current curve estimated by QBOA is generated and contrasted with that of the measured data for the integral and fractional PV model in Figure 7 and Figure 8, respectively. Additionally, the absolute error curves between the measured and estimated load current curves are given in Figure 9 and Figure 10. The visual comparisons validate the efficiency of the proposed QBOA in the identification of the dynamic PV model parameters.

Table 7
Identified parameters of integral dynamic PV model

Algorithm	R_c	C	L	RMSE
GBO	5.624748753	8.16E-06	7.47 E-06	0.008493067
AEO	5.624748647	8.16E-06	7.47 E-06	0.0084931
JS	5.624749	8.15726 E-06	7.47323 E-06	0.008493067
BOA	7.314974895	3.81307E-07	7.3251E-06	0.0084805
proposed QBOA	7.314974219	3.81307E-07	7.3251E-06	0.0084505

Table 8
Identified parameters of fractional dynamic PV model

Algorithm	R_c	C_α	L_β	α	β	RMSE
GBO	5.00598	5.04 E-06	1.35 E-05	1.026120535	0.957165925	0.0082360
AEO	4.55020	1.46 E-05	1.73 E-05	0.917230623	0.940654537	0.0081960
JS	4.69892	4.08 E-05	1.44 E-05	0.833404373	0.953192785	0.007995872
BOA	3.91281E-05	1.80215E-06	7.20724E-05	0.947239708	0.846072368	0.006168565
proposed QBOA	1.00E-05	1.83738E-06	6.60382E-05	0.94299238	0.852227908	0.006155832

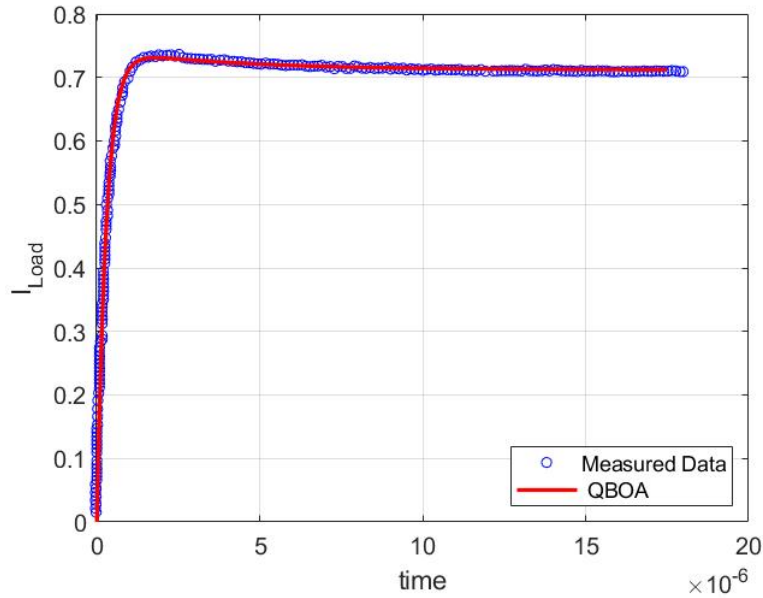


Figure 7. Integral dynamic PV model load current fitting

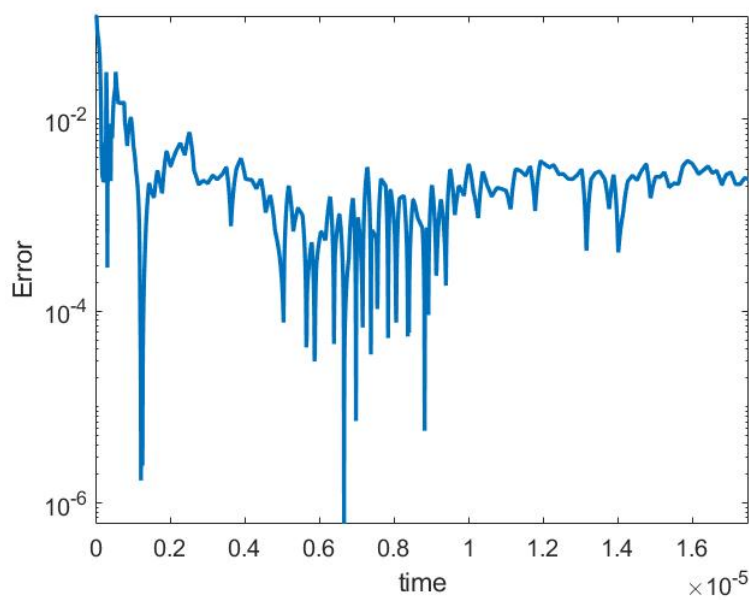


Figure 8. Integral dynamic PV model absolute error

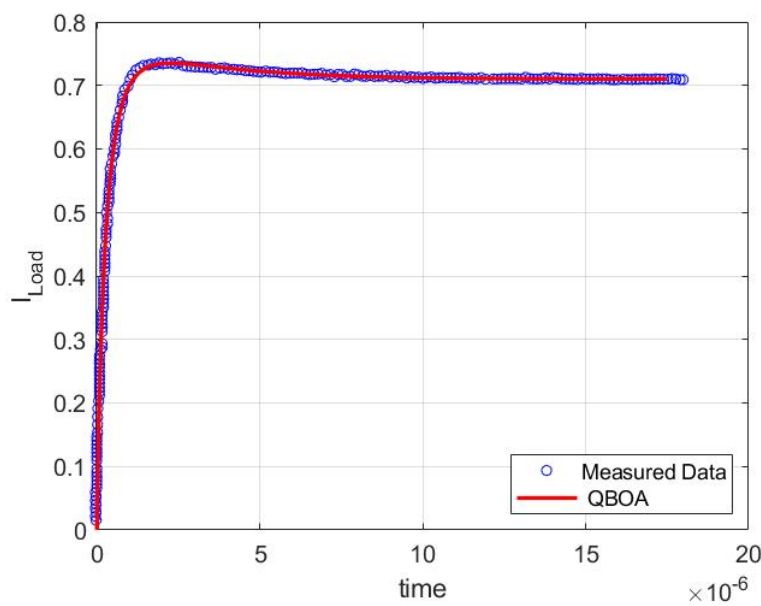


Figure 9. Fractional dynamic PV model load current fitting

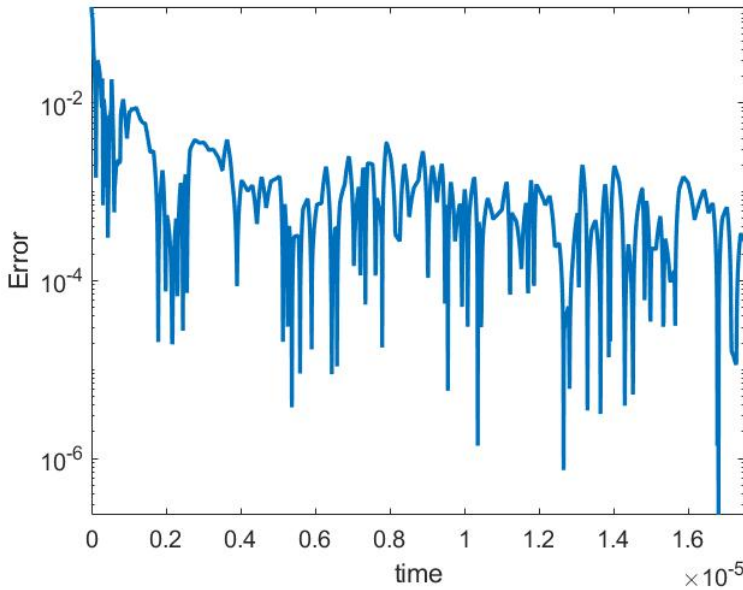


Figure 10. Fractional dynamic PV model absolute error

6. Conclusion

The finite supply of non-renewable resources combined with the world population's rapid increase is driving up demand for energy. Due to this circumstance, there is a risk of environmental pollution and climate change. Therefore, researchers' attention towards renewable energy sources, especially solar energy, have taken the spotlight. Several photovoltaic models have been proposed, where designing a high performance photovoltaic system requires addressing the problem of simulating a solar module and identifying its parameter. This paper employs a quantum behaved meta-heuristic named QBOA for addressing various optimization problems and dynamic photovoltaic models parameter identification. QBOA simulates the reproductive strategies and social behavior of bonobos. Quantum mechanics incorporated into the QBOA algorithm to direct the search agents through the search space. Correspondingly, the exploitation potential of the proposed QBOA algorithm is promoted by this integration. To determine the robustness and coherence of the QBOA, its performance has been examined using the CEC2019 and CEC2005 benchmarks. Additionally, the proposed QBOA is presented to optimize dynamic photovoltaic model's parameter. From the experimental and simulations results, it can be designated that the quantum behaved QBOA sustains the competitiveness on solving different optimization problems and optimizing dynamic photovoltaic models.

References

- [1] AbdelAty A.M., Radwan A.G., Elwakil A.S., Psychalinos C.: Transient and Steady-State Response of a Fractional-Order Dynamic PV Model Under Different Loads, *Journal of Circuits, Systems and Computers*, vol. 27(02), 1850023, 2018. doi: 10.1142/S0218126618500238.
- [2] Ahmadianfar I., Bozorg-Haddad O., Chu X.: Gradient-based optimizer: A new metaheuristic optimization algorithm, *Information Sciences*, vol. 540, pp. 131–159, 2020. doi: 10.1016/j.ins.2020.06.037.
- [3] Ali M.Z., Awad N.H., Suganthan P.N., Duwairi R.M., Reynolds R.G.: A novel hybrid Cultural Algorithms framework with trajectory-based search for global numerical optimization, *Information Sciences*, vol. 334–335, pp. 219–249, 2016. doi: 10.1016/j.ins.2015.11.032.
- [4] Ayang A., Wamkeue R., Ouhrouche M., Djongyang N., Salomé N.E., Pombe J.K., Ekemb G.: Maximum likelihood parameters estimation of single-diode model of photovoltaic generator, *Renew Energy*, vol. 130, pp. 111–121, 2019. doi: 10.1016/j.renene.2018.06.039.
- [5] Chou J.S., Truong D.N.: A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean, *Applied Mathematics and Computation*, vol. 389, 125535, 2021. doi: 10.1016/j.amc.2020.125535.
- [6] Das A.K., Pratihar D.K.: A new Bonobo Optimizer (BO) for real-parameter optimization. In: *2019 IEEE Region 10 Symposium (TENSYP)*, pp. 108–113, 2019. doi: 10.1109/tensymp46218.2019.8971108.
- [7] De Waal F.B.: Bonobo Sex and Society, *Scientific American*, vol. 272, pp. 82–88, 1995. doi: 10.1038/scientificamerican0395-82.
- [8] Di Piazza M.C., Luna M., Vitale G.: Dynamic PV Model Parameter Identification by Least-Squares Regression, *IEEE Journal of Photovoltaics*, vol. 3, pp. 799–806, 2013. doi: 10.1109/jphotov.2012.2236146.
- [9] Di Piazza M.C., Vitale G.: Photovoltaic field emulation including dynamic and partial shadow conditions, *Applied Energy*, vol. 87, pp. 814–823, 2010.
- [10] 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*, pp. 39–43, Nagoya, Japan, 1995.
- [11] Eid H.F., Abraham A.: Solving unconstrained, constrained optimization and constrained engineering problems using reconfigured water cycle algorithm, *Evolutionary Intelligence*, vol. 16, pp. 633–649, 2023. doi: 10.1007/s12065-021-00688-6.
- [12] Eid H.F., Muda A.K.: Adjustive Reciprocal Whale Optimization Algorithm for Wrapper Attribute Selection and Classification, *International Journal of Image, Graphics and Signal Processing*, vol. 11, pp. 18–26, 2019. doi: 10.5815/ijigsp.2019.03.03.
- [13] El-Fergany A.: Efficient tool to characterize photovoltaic generating systems using mine blast algorithm, *Electric Power Components and Systems*, vol. 43(8–10), pp. 890–901, 2015. doi: 10.1080/15325008.2015.1014579.

- [14] Et-torabi K., Nassar-eddine I., Obbadi A., Errami Y., Rmailly R., Sahnoun S., El fajri A., Agunaou M.: Parameters estimation of the single and double diode photovoltaic models using a Gauss–Seidel algorithm and analytical method: A comparative study, *Energy Conversion and Management*, vol. 148, pp. 1041–1054, 2017. doi: 10.1016/j.enconman.2017.06.064.
- [15] Gil-Arias O., Ortiz-Rivera E.I.: General purpose tool for simulating the behavior of PV solar cells modules and arrays. In: *2008 11th Workshop on Control and Modeling for Power Electronics*, pp. 1–5, 2008. doi: 10.1109/compel.2008.4634686.
- [16] Go S.I., Choi J.H.: Design and Dynamic Modelling of PV-Battery Hybrid Systems for Custom Electromagnetic Transient Simulation, *Electronics*, vol. 9, 1651, 2020. doi: 10.3390/electronics9101651.
- [17] Holland J.H.: Genetic algorithms, *Scholarpedia*, vol. 7(12), 1482, 2012. doi: 10.4249/scholarpedia.1482.
- [18] Levin F.S.: *An introduction to quantum theory*, Cambridge University Press, 2002.
- [19] Liang J.J., Suganthan P.N., Deb K.: Novel composition test functions for numerical global optimization. In: *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*, pp. 68–75, 2005. doi: 10.1109/SIS.2005.1501604.
- [20] Mirjalili S.: Moth-flame optimization algorithm: a novel natureinspired heuristic paradigm, *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015. doi: 10.1016/j.knosys.2015.07.006.
- [21] Mirjalili S.: Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete and multi-objective problems, *Neural Computing and Applications*, vol. 27, pp. 1053–1073, 2016. doi: 10.1007/s00521-015-1920-1.
- [22] Mirjalili S., Gandomi A.H., Mirjalili S.Z., Saremi S., Faris H., Mirjalili S.M.: Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017. doi: 10.1016/j.advengsoft.2017.07.002.
- [23] Mirjalili S., Lewis A.: The whale optimization algorithm, *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016. doi: 10.1016/j.advengsoft.2016.01.008.
- [24] Pierozan J., Dos Santos Coelho L.: Coyote optimization algorithm: a new meta-heuristic for global optimization problems. In: *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2633–2640, Brazil, Rio de Janeiro, 2018. doi: 10.1109/cec.2018.8477769.
- [25] Price K.V., Awad N.H., Ali M.Z., Suganthan P.N.: *Problem Definitions and Evaluation Criteria for the 100-Digit Challenge Special Session and Competition on Single Objective Numerical Optimization*, Technical report, Nanyang Technological University, Singapore, 2018.
- [26] Radwan A.G., Salama K.N.: Fractional-order, RC and and RL circuits, *Circuits, Systems, and Signal Processing*, vol. 31, pp. 1901–1915, 2012. doi: 10.1007/s00034-012-9432-z.

- [27] Rashedi E., Nezamabadi-Pour H., Saryazdi S.: GSA: a gravitational search algorithm, *Information Sciences*, vol. 179(13), pp. 2232–2248, 2009. doi: 10.1016/j.ins.2009.03.004.
- [28] Ridha H.M., Heidari A.A., Wang M., Chen H.: Boosted mutation-based Harris hawks optimizer for parameters identification of single-diode solar cell models, *Energy Conversion and Management*, vol. 209, 112660, 2020. doi: 10.1016/j.enconman.2020.112660.
- [29] Tossa A.K., Soro Y.M., Azoumah Y., Yamegueu D.: A new approach to estimate the performance and energy productivity of photovoltaic modules in real operating conditions, *Solar Energy*, vol. 110, pp. 543–560, 2014. doi: 10.1016/j.solener.2014.09.043.
- [30] Zhao W., Wang L., Zhang Z.: Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm, *Neural Computing & Applications*, vol. 32, pp. 9383–9425, 2020. doi: 10.1007/s00521-019-04452-x.

Affiliations

Heba F. Eid

Al-Azhar University, Faculty of Science, Cairo, Egypt, heba.fathy@azhar.edu.eg

Erik Cuevas

University of Guadalajara, Department of Electronics, Mexico, erik.cuevas@cucei.udg.mx

Received: 30.07.2023

Revised: 2.04.2024

Accepted: 2.04.2024

Information for Authors

We accept only the original scientific papers prepared in English. The papers are to be prepared using the LaTeX system. Submitted papers will be refereed by independent reviewers and, if necessary, the Authors may be asked to revise their manuscripts. Proofs will be sent to the Authors for corrections. There is no publication fee. Authors of the accepted papers are eligible to get one hardcopy of the volume containing their contribution free of charge. No postage charges apply.

Our website

<https://journals.agh.edu.pl/csci/>

Open access

This is an open access journal which means that all content is freely available without charge to the user or his/her institution. This is in accordance with the Budapest Open Access Initiative definition of open access. All printed volumes may be accessed at our website.

Indexing

Google Scholar

<http://scholar.google.com>

Index Copernicus

<http://indexcopernicus.com/>

Directory of Open Access Journals

<http://www.doaj.org>

Open Archives Initiative

<http://www.openarchives.org>

Digital Libraries Federation

<http://fbc.pionier.net.pl/owoc/>

BazTech

<http://baztech.icm.edu.pl>

Worldcat

<http://www.worldcat.org>

WorldWideScience.org

<http://worldwidescience.org>

Sherpa Romeo

<http://www.sherpa.ac.uk/romeo/>

Journal TOCs

<http://www.journaltocs.ac.uk>

SCOPUS

www.scopus.com

Web of Science – Emerging Sources Citation Index

www.webofknowledge.com



The Computer Science Journal is published by the AGH University of Science and Technology, Krakow Poland. The Editors of the Journal are members of the Faculty of Computer Science, Electronics and Telecommunications and the Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering. The Editorial Board consists of many renowned computer science researchers from all over the world.

The first issue of the Journal was published in 1999. Currently, the Journal is published quarterly, with the main goal to create a forum for exchanging research experience for scientists specialized in different fields of computer science.

Original papers are sought concerning theoretical and applied computer science problems. Example areas of interest are:

- ☐ theoretical aspects of computer science,
- ☐ pattern recognition and processing,
- ☐ evolutionary algorithms,
- ☐ neural networks,
- ☐ database systems,
- ☐ knowledge engineering,
- ☐ automatic reasoning,
- ☐ computer networks management,
- ☐ distributed and grid systems,
- ☐ multi-agent systems,
- ☐ multimedia systems and computer graphics,
- ☐ natural language processing,
- ☐ soft-computing,
- ☐ embedded systems,
- ☐ adaptive algorithms,
- ☐ simulation.

Abstracts, full versions of the issued volumes and instructions for authors and reviewers may be found at
<http://csci.agh.edu.pl>

