

Minimizing customer waiting time in drone delivery systems: An optimization approach considering heterogeneous fleets and package setup time using modified coot algorithms**Murat Şahin^{a*}**^a*Manisa Celal Bayar University, Turkey***CHRONICLE***Article history:*

Received May 24 2025

Received in Revised Format

July 2 2025

Accepted September 4 2025

Available online September 4
2025

Keywords:

*Drone delivery problem**Coot optimization algorithm**Mathematical modelling**Drone routing with energy
restrictions**Customer waiting time**minimization***ABSTRACT**

This study addresses the drone delivery problem with a unique focus on minimizing total customer waiting times, considering the heterogeneous nature of drones and the setup times required for loading customer demands. Unlike traditional routing problems that prioritize cost and route optimization, this research emphasizes timely deliveries, which are critical in both commercial and humanitarian applications. The study introduces two mathematical models and four versions of the coot optimization algorithm, including three modified variants and one classical version. These algorithms incorporate new movement mechanisms, enhanced leader selection strategies, and adaptations of the regenerating strategy to efficiently solve the drone delivery problem. Computational experiments reveal that one modified coot optimization algorithm significantly outperforms the classical version, offering valuable insights into both coot optimization literature and the drone delivery problem. By emphasizing the importance of timely deliveries, this research provides effective solution strategies applicable to both commercial and humanitarian contexts.

© 2026 by the authors; licensee Growing Science, Canada

1. Introduction

In transportation systems, heavy-load vehicles, such as trucks, lorries, and trailers play a critical role. Although unmanned aerial vehicles (UAVs) cannot match the carrying capacity of these road vehicles, their use in logistics is expected to grow due to several advantages. Drone delivery (DD) enables faster deliveries, shorter travel distances, cost savings, reduced environmental impact, and greater accessibility. With advances in battery technology and payload capacity, DD is projected to become a significant component of transportation systems (Dorling et al., 2017).

Technological advancements in recent decades have expanded UAV use in logistics, including military operations, humanitarian aid, agricultural activities and delivery systems of major companies (Raina et al., 2022; Kim and Kim, 2012). By consuming less energy than traditional vehicles such as trucks, UAVs can need lower transportation costs and reduce delivery times. Their cost-effectiveness makes them well suited for transporting small, portable products over short distances. Consequently, drones are expected to play an increasingly prominent role in future delivery systems (Macrina et al., 2020).

DD involves drones to transport packages from depots to customers, similar to traditional mail trucks or courier services. In routing problems, the primary goal is generally to minimize total energy costs or the distance traveled by vehicles. However, on-time delivery can take precedence over these objectives. The rise of online shopping has increased customer expectations for delivery speed. Quick deliveries have become crucial for customer satisfaction, given the many alternatives available in online shopping. Consequently, businesses have started using drones to deliver orders more quickly and at lower cost. Large companies like Amazon, FedEx, UPS, Posti, and Google have begun employing drones to deliver orders within certain weight and distance limits (Poikonen & Golden, 2020).

Timely delivery of drone-carried packages is essential not only commercially but also for health and humanitarian needs, particularly during natural disasters. Drones can transport critical health supplies that need rapid delivery, such as blood,

* Corresponding author

E-mail sahin.murat@cbu.edu.tr (M. Şahin)

ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)

2026 Growing Science Ltd.

doi: 10.5267/j.ijiec.2025.9.002

vaccines, pharmaceuticals, first aid equipment, and medical samples (Balasingam, 2017). By flying over traffic congestion and taking more direct routes, drones have the potential to significantly reduce delivery times. The implementation of DD systems designed to minimize customer waiting times presents unique challenges. Many studies often address the flying assistant traveling salesman problem by allowing delivery trucks to deploy drones, relying on a combined fleet of trucks and drones for deliveries. However, with continuous advancements in battery technology and increasing drone carrying capacity, DD technology is expected to play significant role in transportation systems due to its numerous advantages (Dorling et al., 2017).

Advancements in battery technology and energy management have enhanced delivery efficiency by reducing the need for frequent recharging. Effective route planning requires seamless coordination between customer order planning and drone selection. This study addresses the drone delivery problem (DDP) with heterogeneous drones to minimize total customer waiting times. Most routing problems in the literature, whether involving vehicles or drones, focus on minimizing routes and costs. In DDPs, given the relatively low transportation cost associated with drones, the priority can shift toward on-time deliveries. Factors affecting delivery time include drone speed, payload capacity, battery capacity, and package loading time. There are few studies on drone routing problems that explicitly aim to minimize total customer waiting time. By emphasizing this objective, the study aligns with the needs of sectors where prompt delivery is crucial, such as e-commerce and emergency medical supply distribution (Akincilar & Akincilar, 2017). Addressing drone heterogeneity provides a more realistic and practical solution framework. The findings suggest considerable commercial and humanitarian benefits, including improved customer satisfaction and more effective disaster response, while also advancing the optimization literature with novel algorithmic strategies. To the authors' knowledge, no prior study addresses the heterogeneous DDP considering the setup time required to load packages into drones with aim of minimizing the total customer waiting times. This research suggests four different metaheuristics based on the recently developed coot optimization algorithm (COA) after giving the two different mathematical models of the problem. Three modified versions and one classical version of COA are introduced by altering its movement, leader selection, and regenerating mechanisms. Computational experiments show that one of the COA uses the suggested movement, leader selection, and regenerating mechanisms has much better performance than the classical version. Overall, this study makes a notable contribution to both the COA and DDP literature, by defining the problem with 2 different mathematical models and applying more effective movement and leader selection mechanism in COA framework.

The rest of the article is organized as follows: Section 2 presents the literature review related the handled problem, Section 3 defines the problem in detail, and Section 4 provides two different linear integer programming formulations. Section 5 describes the classical and modified COAs, while Section 6 reports the computational experiments conducted using the dataset generated in this study. Finally, Section 7 explains the conclusions and future research directions.

2. Literature Review

Many variants of the DDP have emerged by varying the type of vehicle used (only drone, or drone and truck), the number of depots (one or many), drone attributes (homogeneous or heterogeneous), technical constraints, and optimization objectives. This field has expanded significantly since Murray and Chu (2015) introduced the concept of combining drones with vehicles in transportation, known as the traveling salesman problem with a “sidekick,” where the sidekick is a drone carried on a delivery vehicle. Since then, numerous studies (Poikonen and Golden, 2020; Moshref et al., 2020; Luo et al., 2021; Huang et al., 2022; Lu et al., 2022; Kuo et al., 2022; Yin et al., 2023; Meng et al., 2023; Tomas et al., 2023; Sadok et al., 2024, etc.) have examined such problems. Research on drones indicates that, with advancing technology, drone transportation is expected to play a major role in logistics without requiring support from additional vehicles. DDP can be defined as a new variant of the vehicle routing problem that involves a fleet composed solely of drones. Unlike the vehicle routing problem, the DDP includes factors specific to drones, such as limited flight time, restricted battery capacity, and varying energy consumption models. The problem discussed in this article is closely related to both vehicle routing and DD challenges. Considering the objective functions and the characteristics of the problem, this section reviews studies focusing on customer waiting times and drone delivery challenges. For readers seeking more detailed information about the literature, the studies presented by Macrina et al. (2020), Thibbotuwawa et al. (2020), Javadi and Winkenbach (2021), and Jazairy et al. (2024) are recommended. A summary of the reviewed literature is presented in Table 1.

Drone delivery and routing problems: Dorling et al. (2017) consider DDP in which drones can serve more than one customer per route and can perform more than one trip. They suggest an energy consumption model as a function of battery and payload weights, and present a mathematical model beside the simulated annealing algorithm as a solution method. Coelho et al. (2017) suggest a multi-objective DDP and introduce charging stations to address challenges related to limited drone flying range. They employ a rate that depends on the drone’s speed as a decision variable, and evaluate the energy required for recharging at charging stations, providing both a mathematical formulation and a metaheuristic approach. Shavarani et al. (2018) deal with a distance-constrained mobile hierarchical facility location problem to find the optimal number of locations, launch and recharge stations with the aim of minimizing the total costs of the system that includes establishment cost for launching/recharging stations, drone procurement, and drone usage costs. They solve the stated problem by providing a mathematical model and a genetic algorithm based heuristic. Torabbeigi et al. (2019) define the design of a parcel delivery system using drones, which involves the strategic planning of the system. The amount of payload directly affects the battery consumption rate, which can cause a disruption in the delivery of goods. Two optimization planning models

are proposed to design drone-based parcel delivery: strategic and operational planning. The optimal solution suggests the minimum number of drones required to serve all customers. The variable preprocessing technique, primal and dual bound generation schemes are enhanced to reduce computational time.

Table 1
Summary of the literature survey

Reference	Problem Type	Objective	Method	Drone Type	Contribution
Dorling et al. (2017)	DDP	Total delivery time and total cost	MM and heuristic	H	New energy consumption model
Coelho et al. (2017)	DDP	Total distance, number of used drones, makespan and etc.	MM and heuristic	H	Integrating drones into the new concepts of mini/microgrid systems, in which vehicles can be charged at different points
Troudi et al. (2018)	DDP	Distance, number of drones and battery minimization.	MM	H	Creating a decision-making tool for the design of a drone fleet in the case of forecast deliveries over a time horizon under operational constraints
Liu (2019)	Dynamic DDP	Distance, number of drones, batteries	MM	Ht	Proposes an optimization-driven, progressive algorithm for online fleet dispatch operations
Wen and Wu (2022)	DDP	Distance, delivery time	MM and heuristic	Ht	A new two-echelon routing problem based on heterogeneous multi-drone, routing problem is handled
Phalapanyakoon and Siripongwutikorn (2021)	UAV Routing	Recharging cost, travel distance, used UAV and timely service	MM	H	The electric cost incurred by UAV recharging proportional to actual flying distances is incorporated
Cheng et al. (2020)	DDP	Total cost minimization	MM and B&C	H	Nonlinear energy consumption model and suggests exact solution algorithm
Zhang et al. (2021)	DDP	Giving framework for energy consumption model	-	-	Review for energy consumption model
Shavarani et al. (2018)	DDP	Total cost minimization	MM and GA	H	A realistic mathematical model and effective solution method
Torabbeigi et al. (2019)	DDP	Minimization the number of drones used	MM	H	Providing the solution methodology consisting of a variable preprocessing technique, primal and dual bound generation
Moshref et al. (2020)	T-DDP	Minimization the total customer waiting times	MM and heuristic	H	Minimizing the total customer waiting times in a T-DDP
Luo et al. (2021)	T-DDP	Minimizing the delivery time	MM and heuristic	H	Using the multi visit drone in a T-DDP
Kuo et al. (2022)	T-DDP	Minimize the total travelling cost	MM and heuristic	H	T-DDP with time windows
Meng et al. (2023)	T-DDP	Minimizing total cost	MM and heuristic	H	New variant of the T-DDP contains pick up and delivery systems
Yin et al. (2023)	T-DDP	Minimizing total cost	MM and B&P and B&C	H	T-DDP with time windows and exact solution algorithm
Bruni et al. (2023)	L-DDP	Minimizing the customer waiting times	MM and B&Ch	H	Introduce a combined location and routing problem in last-mile delivery

DDP :drone delivery problem, T-DDP :truck-drone combined delivery problem, L-DDP: location assignment and drone delivery problem, MM: mathematical model, B&P: branch and price algorithm, B&C: branch and cut algorithm, B&Ch: branch and check algorithm, GA: genetic algorithm, H: homogeneous, Ht: heterogeneous.

In the variant suggested by Troudi et al. (2019), drones can perform more than one mission per day and multiple visits. For computing energy consumption during a mission, they propose an approximation model similar to that of Dorling et al. (2017). Their objective is to minimize the total travelled distance, the total number of required drones, and the total number of required batteries. Liu (2019) deals with an on-demand meal delivery system and suggests a dynamic DD model to optimize it. He develops both a static and a dynamic model, considering several constraints and assumptions pertinent to food delivery. The model accounts for real-world issues, such as the uncertainty of order locations and sizes, battery usage, and battery swapping. The objective function aims to ensure safety, reduce the total lateness, and enhance efficiency. Cheng et al. (2020) consider a multi-trip drone routing problem, where drones' energy consumptions are formulated as a nonlinear function of travel distance and payload weight. They suggest a mathematical model and a branch-and-cut based exact solution algorithm for minimizing the total cost. Zhang et al. (2021) suggest a uniform framework for different drone energy consumption formulations and the inter-relationships between key factors. Luo et al. (2021) consider the multi-visit traveling salesman problem with multi-drones, whose aim is to minimize the makespan required by the truck and the drones to serve all customers together. They present a mathematical model of the problem and a tabu search based metaheuristic for the solution. Phalapanyakoon and Siripongwutikorn (2021) handle the problem of route planning for rechargeable UAVs under the mission time constraint in

cases where more than one trip per round is required due to limited battery capacities. They try to detect the number of UAVs to be deployed and the flying paths that minimize the total mission cost by developing a mathematical model. Wen and Wu (2022) deal with heterogeneous DDP for parcel delivery in which there are two types of drones called mother and small drones. Mother drones carry small drones and small drones are used to deliver parcels. They aim to minimize the delivery time and cost by developing a mathematical model and a heuristic. Meng et al. (2023) consider a different variant of the combined truck-drone routing problem, which allows drones to serve more than one customer per dispatch and provide both delivery and pickup services in each flight. In the stated problem homogenous trucks are equipped with only one drone and each drone can visit multiple customers as long as battery restriction is satisfied. They suggest a linear programming formulation of the problem and improve a heuristic for the solution of the problem.

Recent routing problems about delivery time: Recent studies on routing problems increasingly consider delivery time as a critical factor. Moshref et al. (2020) develop a mathematical formulation and a heuristic for the planning of routes in a multi-model delivery system comprising trucks and UAVs. They aim to minimize total customer waiting times by synchronizing drones and trucks. The flying range of a drone is limited by time and one drone can only serve one customer request per dispatch. Kuo et al. (2022) consider drone and truck cooperation by developing a model for the truck-drone delivery problem that takes into account the time windows. A mathematical model and a heuristic based on the variable neighborhood search algorithm are proposed to minimize the total travelling cost. Yin et al. (2023) address the truck-based DD routing problem with time windows. In the stated article a set of trucks and drones collaborate to serve customers. A drone can take off from its associated truck at a location, independently serve one or more nodes within the time windows, and return to the truck at another location along the truck route. They try to minimize the total cost by using branch & price and branch & cut algorithms in addition to the mathematical model. Bruni et al. (2023) consider the location and routing for DDP with the aim of finding the optimal subset of fulfillment centers to use as drone launching and landing platforms and the optimal drone routes to minimize the sum of customers' waiting times. After presenting the mathematical model of the problem, they suggest a branch and check based exact solution algorithm as a solution method. Şahin (2023) suggests a constructive heuristic for the heterogeneous DDP that considers packages' setups and battery capacity with the aim of minimizing the weighted total waiting times of customers.

Coot optimization algorithm: COA was first presented by Naruei and Keynia (2021) for solving continuous optimization problems, inspired by the movements of coot birds, and compared against many existing metaheuristics. Due to its good performance COA has started to be applied for solving many different hard problems in various areas. Abdulsahab and Kadhim (2022) improved a COA for robot path planning in unknown environments. The suggested COA tried to imitate the real world by adding the mobile robot's actual size and the kinematic model with specifications for mobile robots. Kien et al. (2022) used COA for the optimal placement of photovoltaic generators in distribution systems considering variation in load and solar radiation. Karimunnisa and Pachipala (2023) classified the tasks and schedules by applying an enhanced COA in cloud computing. Özden and İşeri (2023) used COA for optimizing artificial neural network parameters to increase the effectiveness of the artificial neural network model. Aslan and Koç (2024) suggested a modified COA for solving the community detection problem in social networks. The COA, which has had successful applications in many areas in recent years, was also used to solve the DDP defined in this study.

Literature survey shows that studies on DDP have increased significantly in recent years. Although minimizing cost and route length is adopted as the main goal in routing problems, it is of great importance to fulfill customer requests within the planned time. This article deals with total lateness and detailed information about the handled problem is given in the following section.

3. Problem Definition

This article explores methods to determine optimal routes for drones in order to minimize total customer waiting time, considering the required loading time (setup time) to load packages into drones. The problem involves determining which drones will be assigned to specific customer packages and the sequence in which each drone will deliver these packages. Essentially, there are two decision variables: assigning packages to drones and determining the routes for each drone. Customer packages need to be delivered by a specified *due time* (dt_i). If a drone delivers a package to customer i later than dt_i , the lateness value is added to the objective function. Lateness for customer i (denoted as l_i) is calculated using the equation $l_i = \max(0, st_i - dt_i)$, where st_i is the serving time for customer i . The drone fleet is heterogeneous, each drone has various carrying capacity (both in units and weight), speed, and battery capacity. There are a certain number of customer requests, each with specific delivery times and package weights. Drones can serve multiple customers as long as their weight, quantity, carrying and battery limits permit. According to Dorling et al. (2016), there is an almost linear relationship between a drone's battery capacity and airtime, depending on the load it carries. Therefore, we assume that battery consumption is related to the payload the drone carries while in the air. Additionally, to make the problem more realistic, a setup time for loading customer packages into drones is considered. Drone deliveries can be cost-effective compared to road-based methods. However, the focus here is on the timely delivery of customer demands rather than route and cost minimization. Thus, the sole objective of this study is to minimize the total lateness in delivering customer demands. The remaining assumptions of the problem are outlined below.

- There is only one of each type of drone.

- Each drone has different carrying capacities in terms of quantity and weight.
- Each drone has different battery capacity and speed.
- The service times of customers can be neglected compared to the flying time.
- Distances between customer locations are calculated using Euclidean distance.
- A setup time for loading customer packages into drones is required, and this time is not related to the specific drone.
- The required loading time for packages into drones does not affect battery capacity.
- There is a linear relationship between battery capacity, the payload carried, and the time the drone stays in the air.
- Each drone starts at the depot and can only take one tour but can serve multiple customers.
- Assignment of packages to drones and routing is based on the technical attributes of drones, such as carrying capacity, battery capacity, and load capacity.

Consider an example with three drones and six customers, resulting in seven locations including the depot. Table 2 provides detailed information about the drones and customers. The Euclidean distances between the customer locations can be easily computed using their x and y coordinates. In the stated table, γ represents a constant value related to the battery capacity that the drone consumes depending on the payload and the time it stays in the air.

Table 2

Information about the example.

(a) Information about the drones						
Drone ID	Unit Capacity (uc_k)	Weight Capacity (wc_k)	Average Velocity (v_k)	Battery Capacity (bc_k)	Energy Consumption	Drone Weight
1	3	1	300	4000	$3*\gamma$	3.5
2	3	0.9	275	5000	$4*\gamma$	5.5
3	2	2.0	250	6000	$5*\gamma$	6.0

(b) Information about the customers					
Customer ID	$X_{Coordinate}$	$Y_{Coordinate}$	Package Weight (pw_i)	Due Time (dt_i)	Package Setup Times ($Cset_i$)
0 (Depot)	1500	1500	-	-	-
1	1750	1300	0.44	2	0.15
2	1655	1453	0.38	2	0.22
3	1453	1041	0.60	5	0.20
4	1881	1071	0.25	3	0.16
5	1005	1820	0.34	2	0.33
6	1554	1608	0.58	4	0.14

An illustrative example of drone assignments and route planning is provided in Figure 1. In this figure, packages 5 and 6 are assigned to drone 1, and the route for this drone is 0-5-6-0. When making these assignments and planning the route, the weight, unit, and battery capacities of drone 1 are checked. The unit capacity of drone 1 is 3, and two packages are assigned. The weight capacity of drone 1 is 1 kg, and the total weight of packages 5 and 6 ($0.34 + 0.58$) kg is less than 1 kg.

To control the battery capacity, we need the Euclidean distances from 0 to 5, 5 to 6, and 6 to 0, which are approximately 589.43, 588.51, and 120.75 units, respectively. The travel times for these segments are approximately 1.963, 1.962, and 0.403 time units, respectively. The required setup time to load the packages into drone 1 is calculated as 0.47 time units ($0.33 + 0.14$). It should be noted that packages 5 and 6 are loaded into drone 1 at the depot.

In this route, drone 1 first flies from the depot to location 5 in 1.963 time units with a payload of 0.92 kg ($0.58 + 0.34$) payload. The required battery capacity for drone 1 to fly from the depot to location 5 is calculated by multiplying the total weight (the weight of the packages and the drone) by flying duration and the constant γ . Similarly, the required battery capacity for flying from location 5 to location 6 and from location 6 to the depot is approximately 24.01γ and 4.23γ , respectively. The total required battery capacity for drone 1 is 54.27γ .

For drone 1 to complete the 0-5-6-0 tour without running out of battery, its battery capacity must be greater than 54.27γ . If the value of γ is greater than 73.71, the battery capacity of drone 1 would be insufficient for the given assignments. Assuming the battery capacity is sufficient, the arrival times at locations 5 and 6 are computed by considering flying and setup times as $0.47 + 1.963 = 2.01$ and $0.47 + 1.963 + 1.962 = 4.395$, respectively.

For other drones, whether they can complete their tours is calculated similarly, depending on their technical specifications. Once the service time and due time of the customers are known, the objective function value can be easily calculated. An illustration is provided in Figure 1, and a more detailed example is available in Section 5.2, where the encoding scheme and

solution-building algorithm are explained. In the following sections, two mathematical models and four COA approaches are presented for solving the addressed problem.

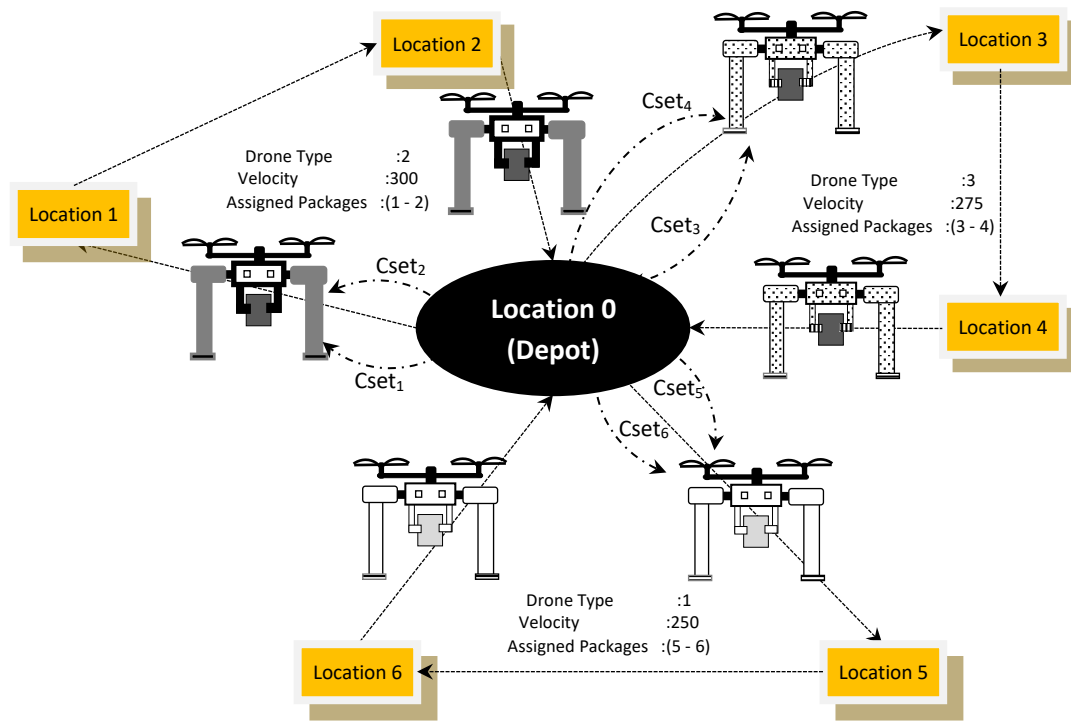


Fig. 1. An illustrative for heterogeneous drone routing.

4. Mathematical Models

The problem is formulated using two different modeling approaches. The first model is based on the approach proposed by Dorling et al. (2016). In this model, routes and drone assignments are represented using a binary variable set with three indices (x_{ijk}). x_{ijk} takes the value 1 if drone k goes to node j after visiting node i . This approach determines which customers will be visited by which drone and the specific route taken. The other variables used in the mathematical model are detailed below. Considering all variables, the upper bound for the number of variables is $O(n^3)$. This is primarily because route and drone assignments are determined within a single set of variables. The second model separates drone assignments and route assignments into two distinct variables, aiming to reduce the upper bound of the number of variables to $O(n^2)$. The notations and mathematical formulations of the problem are provided below:

First Mathematical Model (Model-1)

1. Notations and sets

- N: Set of locations without depot
 N_0 : Set of locations with depot ($N_0 = N \cup \{1\}$ where $\{1\}$ is the depot node)
K: Set of drones (1, 2, ..., $|K|$)
M: Sufficiently big number
 m_k : The weight of drone k
 C_{Set}^h : Setup time of customer h 's package
 d_{ij} : Total distance from location i to location j
 v_k : Average velocity of drone k during the flying process
 uc_k : Package capacity in unit for drone k
 wc_k : Weight capacity of drone k
 bc_k : Battery capacity of drone k
 q_i : Package weight of location i
 dt_i : Due time of the customer at location i
 φ : Constant showing the amount of milliamps (mA) to stay a drone on air for a unit time depending on the payload weight.

2. Variables

- x_{ijk} : Binary variable with value 1 if drone k goes to from location i to location j .
 y_{ik} : Continuous variable which shows the payload weight of drone k at location i .

- f_{ik} : Energy consumed from the battery of drone k for reaching location i .
- l_i : Lateness of package delivery for location i .
- t_i : The service time for location i .

3. Objective function and constraints

$$\min \sum_{i \in N} l_i \tag{1}$$

Subject to

$$\sum_{k \in K} \sum_{\substack{i \in N_0 \\ i \neq j}} x_{ijk} = 1 \quad \forall j \in N \tag{2}$$

$$\sum_{j \in N} x_{1jk} \leq 1 \quad \forall k \in K \tag{3}$$

$$\sum_{\substack{j \in N_0 \\ i \neq j}} x_{ijk} - \sum_{\substack{j \in N_0 \\ i \neq j}} x_{jik} = 0 \quad \forall i \in N_0 \quad \forall k \in K \tag{4}$$

$$t_i - t_j + d_{ij}/v_k \leq M \cdot (1 - x_{ijk}) \quad \forall k \in K \quad \forall i \in N_0 \quad \forall j \in N \tag{5}$$

$$t_1 - t_j + d_{1j}/v_k + \sum_{l \in N_0} \sum_{h \in N} C_{Set}^h \cdot x_{lhk} \leq M \cdot (1 - x_{1jk}) \quad \forall k \in K \quad \forall j \in N \tag{6}$$

$$\sum_{i \in N_0} \sum_{\substack{j \in N_0 \\ i \neq j}} x_{ijk} \leq uc_k + 1 \quad \forall k \in K \tag{7}$$

$$\sum_{i \in N_0} \sum_{(j \in N) \wedge (j \neq i)} q_j \cdot x_{ijk} \leq wc_k \quad k \in K \tag{8}$$

$$l_i \geq t_i - dt_i \quad \forall i \in N \tag{9}$$

$$y_{ik} \geq y_{jk} + q_j - M \cdot (1 - x_{ijk}) \quad \forall k \in K \quad \forall i \in N_0 \quad \forall j \in N \wedge (i \neq j) \tag{10}$$

$$f_{ik} - f_{jk} + \phi \cdot ((m_k + y_{ik}) \cdot d_{ij}/v_k) \leq M \cdot (1 - x_{ijk}) \quad \forall k \in K \quad \forall i \in N_0 \quad \forall j \in N \wedge (i \neq j) \tag{11}$$

$$f_{ik} + \phi \cdot ((m_k + y_{ik}) \cdot d_{i1}/v_k) - M \cdot (1 - x_{i1k}) \leq bc_k \quad \forall k \in K \quad \forall i \in N_0 \tag{12}$$

$$x_{ijk} \in \{0,1\} \quad \forall i, j \in N_0, \forall k \in K \tag{13}$$

$$y_{ik}, f_{ik} \geq 0 \quad \forall i \in N_0, \forall k \in K \tag{14}$$

$$l_i, t_i \geq 0 \quad \forall i \in N \tag{15}$$

The objective of the mathematical model is defined by Equation (1), which aims to minimize the total waiting time of the customers. Visiting each customer with only one drone and only one time is satisfied by Equation (2). According to Constraints (3), a drone can leave the depot at most once. Constraint (4) dictates that should a drone visit a customer, it must depart from that location; likewise, if it leaves from the depot, it must return there. Constraint (5) arranges the service time among the adjacent travels. If the drone k goes from location i to location j , the service time of location j must be bigger than the sum of service time of location i and the travel time (d_{ij}/v_k). Constraint (6) considers the setup times of customer packages for the related drones. When the drone leaves the depot to arrive at any customer, the total setup time of the customers' packages loaded to that drone should also be added to the time it takes to get there. The unit capacity of the drone is enforced by Constraint (7). Constraint (8) satisfies the capacity restriction according to the weight loaded to the drone. Constraint (9) computes the lateness of each node according to the service and due time. The payload weight of each drone is arranged with Constraint (10). Constraint (11) regulates the energy consumption of the battery according to payload weight. Constraint set (12) satisfies the battery restriction. It checks whether the energy adequacy can be returned to the depot using the given packages and routes. Constraint sets (13-14-15) are integrality and sign constraints for the decision variables.

Second Mathematical Model (Model-2)

4. Variables

- x_{ij} : Binary variable with value 1 if a drone goes to from location i to location j .
- w_{ik} : Binary variable with value 1 if customer i is assigned to drone k .

- wu_k : Binary variable with value 1 if *drone k* is used.
 y_{ik} : Continuous variable which shows the package weight of *drone k* at *location i*.
 f_{ik} : Energy in mA consumed from the battery of *drone k* for reaching *location i*.
 l_i : Lateness of package delivery for *location i*.
 t_i : The time of package service for *location i*.

5. Objective function and equations

$$\min \sum_{i \in N} l_i \quad (16)$$

subject to

$$\sum_{\substack{i \in N_0 \\ i \neq j}} x_{ij} = 1 \quad j \in N \quad (17)$$

$$\sum_{j \in N} x_{Tj} \leq |K| \quad (18)$$

$$\sum_{\substack{j \in N_0 \\ i \neq j}} x_{ij} - \sum_{\substack{j \in N_0 \\ i \neq j}} x_{ji} = 0 \quad \forall i \in N_0 \quad (19)$$

$$\sum_{k \in K} w_{ik} = 1 \quad \forall i \in N \quad (20)$$

$$w_{1k} = 1 \quad \forall k \in K \quad (21)$$

$$wu_k \leq \sum_{i \in N} w_{ik} \quad \forall k \in K \quad (22)$$

$$\sum_{i \in N} x_{1i} = \sum_{k \in K} w_{1k} \quad (23)$$

$$\sum_{i \in N} w_{ik} \leq uc_k \quad \forall k \in K \quad (24)$$

$$\sum_{i \in N} q_i \cdot w_{ik} \leq wc_k \quad \forall k \in K \quad (25)$$

$$\sum_{k \in K} k \cdot w_{ik} - \sum_{k \in K} k \cdot w_{jk} \leq M \cdot (1 - (x_{ij} + x_{ji})) \quad \forall i \in N \quad \forall j \in N \wedge (i \neq j) \quad (26)$$

$$-t_j + t_i + w_{jk} \cdot \frac{d_{1j}}{v_k} + \sum_{i \in N} C_{Set}^i \cdot w_{ik} \leq M \cdot (1 - x_{1j}) \quad (27)$$

$$t_i - t_j + \left(\sum_{k \in K} w_{jk} \cdot \frac{d_{ij}}{v_k} \right) \leq M \cdot (1 - x_{ij}) \quad \forall i \in N_0 \quad \forall j \in N \quad (28)$$

$$t_i - dt_i \leq l_i \quad \forall i \in N \quad (29)$$

$$y_{jk} + q_j - M \cdot (2 - x_{ij} - w_{jk}) \leq y_{ik} \quad \forall k \in K \quad \forall i \in N_0 \quad \forall j \in N \wedge (i \neq j) \quad (30)$$

$$f_{ik} - f_{jk} + \varphi \cdot ((y_{ik} + m_k) \cdot (d_{ij}/v_k)) \leq M \cdot (2 - x_{ij} - w_{jk}) \quad (31)$$

$$\forall k \in K \quad \forall i \in N_0 \quad \forall j \in N \wedge (i \neq j)$$

$$f_{ik} + \varphi \cdot ((y_{ik} + m_k) \cdot (d_{i1}/v_k)) - M \cdot (2 - x_{i1} - w_{ik}) \leq bc_k \quad \forall k \in K \quad \forall i \in N_0 \quad (32)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i, j \in N_0, \quad \forall k \in K \quad (33)$$

$$w_{ik} \in \{0, 1\} \quad \forall i \in N_0, \quad \forall k \in K \quad (34)$$

$$wu_k \in \{0, 1\} \quad \forall k \in K \quad (35)$$

$$y_{ik}, f_{ik} \geq 0 \quad \forall i \in N_0, \quad \forall k \in K \quad (36)$$

$$l_i, t_i \geq 0 \quad \forall i \in N \quad (37)$$

Equation (16) computes the objective function value, which is same as in the previous mathematical model. It aims to minimize the total lateness. Constraint (17) guarantees that each node is visited only once by a drone. Constraint (18) satisfies that the number of drones leaving the depot cannot be exceed the total number of drones. According to Constraint (19), a drone arrived at a node must leave that node again. Each customer's package must be assigned to a drone according to Equation

(20). The availability of all drones in the depot is ensured by constraint (21). Assigning the appropriate value to the variable wu_k is satisfied by Constraint (22). According to constraint (23) the number of used drones must be equal to the number of drones traveling from the depot to the customer. Unit capacity constraint and weight capacity constraint of the drones are satisfied by inequalities 24 and 25. Constraint (26) is used to assign the customer to the same drone that is on the same route. Constraint 27 provides the total preparation time for packages that need to be loaded onto the drone. Constraint (28) regulates the service time between customers on the same route according to the velocity of the drone. The lateness of a customer package is determined by Constraint (29). Constraint (30) computes the payload weight of a drone at a node. The battery consumption of the drones at each node is calculated by Constraint (31). Constraint (32) ensures that the relevant drone returns to the depot on the specified route before its battery runs out. Constraint sets (33-34-35-36-37) are integrality and sign constraints for the decision variables. This mathematical model is referred to as “Model-2” in the rest of the article.

5. Coot Optimization Algorithm

Metaheuristic algorithms have become increasingly effective in solving various optimization problems over the last few decades, often providing solutions when exact methods fail to deliver within acceptable time frames. Moreover, metaheuristics are used effectively in solving various routing problems (Xu et al., 2024; Golmohammadi et al., 2016; Hong et al., 2024). These algorithms draw inspiration from diverse sources and are categorized into different classes. For example, swarm-based algorithms, which mimic the collective behavior of swarms or groups of organisms and include particle swarm optimization, firefly algorithm, artificial bee colony optimization. Evolutionary algorithms, inspired by the principles of biological evolution, include genetic algorithm, genetic programming, evolutionary algorithm, differential algorithm, etc.. Physics-based algorithms simulate physical phenomena to guide the optimization process and consist simulated annealing, gravitational search, curved space optimization, electromagnetic optimization algorithms. Human-based algorithms draw inspiration from human behavior or social systems and include harmony search, teaching-learning-based optimization, and imperialist competitive algorithm. Apart from these, numerous other metaheuristic algorithms exist, each known for its effectiveness in solving various problems. One such algorithm is the COA, inspired by the natural behavior of coot birds, which has shown promise in solving continuous optimization problems (Naruei and Keynia, 2021). Being a relatively recent addition to the field, the COA presents opportunities for further refinement and development. In this study, it is aimed to solve the stated DDP by using different movement methods in the COA. Coots, small water birds, inspired the development of the COA by Naruei and Keynia (2021). These birds exhibit unique behaviors on water surfaces, such as traveling at angles to their direction of motion and avoiding areas that repel other species such as surf scoters. Based on these observations, Naruei and Keynia (2021) identified three primary movement patterns among coots: disordered, synchronized, and chain movements. Building on these natural behaviors, Naruei and Keynia (2021) proposed four distinct motion mechanisms for the COA:

1. Random movement,
2. Chain movement,
3. Adjusting the position based on the group leaders
4. Leading the group by the leaders towards the optimal area.

This study aims to enhance the effectiveness of the COA by refining and modifying the movement mechanisms proposed by Naruei and Keynia (2021). The following section provides detailed information about the components of the COA.

5.1. Coot Optimization Algorithm Components and New Movement Approaches

The COA consists several key components. The process begins with generating a population of coots, from which NL individuals are randomly selected. The best-performing coot in this population is designated as the global leader and directs the movement of the NL individuals. The movements of the coot population are then determined using four distinct approaches, as mentioned earlier. In addition, the algorithm incorporates mechanisms for constructing solutions and defining an encoding scheme tailored to the specific problem at hand. The following subsections describe these components in detail.

5.2. Encoding Scheme and Solution Building

In the suggested COA for the solution of the heterogeneous DDP with package setup times, an encoding scheme using continuous numbers ranging from 0 to 1 is employed. In this encoding scheme, an array of length $(n + m)$ is used, where n represents the number of customers (locations), and m represents the number of drones. These numbers serve as priorities for assigning customers to drones. An illustration of the encoding scheme with six locations and three drones is depicted in Figure 2. In this array, each position corresponds to a coot's position in the COA. The first n numbers in the array denote the selection precedence of each location, while the last m elements indicate the priority values of the drones. This encoding scheme facilitates the representation of potential solutions in the COA for optimizing the DDP.

Location 1	Location 2	Location 3	Location 4	Location 5	Location 6	Drone 1	Drone 2	Drone 3
0.93	0.64	0.88	0.58	0.72	0.23	0.52	0.71	0.58

Fig. 2. An illustration of a coot position

The algorithm for generating a feasible solution from a given encoding scheme and determining its fitness value is provided in Fig. 3. Conducting a computational experiment to demonstrate this algorithm would be beneficial. For instance, consider a DDP contains 6 locations and 3 drones, with parameter values outlined in Table 2. The routes for the drones and the fitness value are computed using the algorithm specified in Fig. 3, applied to the encoding scheme shown in Fig. 2.

Algorithm : Generation of drone routes from a coot's position
Input : Coot's position along with the problem parameters
Outputs : Fitness value of the coot and the routes for drones

Step 1. Set $dr_k = 0$ for $k \in \mathbf{K}$; $dw_k = 0$ for $k \in \mathbf{K}$; $du_k = 0$ for $k \in \mathbf{K}$; $Route_k = \{1\}$, $k \in \mathbf{K}$

Step 2. Generate the sequence of locations (SL) and drones (SD) according to position of coot.

while (SL \neq {})

{

Step 3. Take current location (i) from SL, and find the most appropriate drone considering SD.

Step 3.1. Find the elements of AD set that includes the appropriate drones by controlling the drone unit capacity, drone weight capacity and drone battery capacity

$$AD = \{k \mid (du_k + 1) \leq uc_k, (dw_k + q_i) \leq wc_k, RB(Route_k + i) \leq bc_k\}$$

Step 3.2. *If* (AD = {}) (This means any feasible solution cannot found for the given coot position)

Add a penalty value to the objective function and Go to **Step 6**;

else Select the most appropriate drone (**drone k**) which is ready at the earliest time among the list of AD considering only the travel time. If more than one drones are the most appropriate, select the one that appears earlier in the SD list.

Step 4. Update the $dr_k = dr_k + (d_{ij}/v_k)$ where j is the last location in $Route_k$

$$dw_k = dw_k + q_i;$$

$$du_k = du_k + 1;$$

$$Route_k = Route_k + i$$

$$SL = SL - \{i\}$$

}

Step 5. Compute the fitness value considering setup times of drones.

Step 6. Output the fitness value and the routes for drones.

dr_k : ready times of drone k ; dw_k : total weights of packages which are assigned to drone k ; du_k : the number of the packages which are assigned to drone k ; $Route_k$: route of drone k ; q_i : package weight of location i ; d_{ij} : the distance between location j and i ; v_k : velocity of drone k ; $RB(Route_k + i)$: the required battery capacity for the new route which is formed by inserting the location i to the $Route_k$

Fig. 3. Algorithm for generating a solution and computing the fitness value of a coot's position

In the algorithm's first step, essential parameters and sets are initialized. The second step involves forming sets SL and SD based on the coot positions. For the given encoding scheme coot position indicates SL as {1, 3, 5, 2, 4, 6} and SD as {2-3-1}.

In the first row of Table 3, the list of assignable drones (AD) is found as {2-3-1}, indicating that all drones can reach customer 1 based on their unit, weight, and battery capacities. The most appropriate drones, considering their earliest ready times (where all drones with $dr_k=0$ initially), are {2-3-1}. Drone 2 is chosen for customer 1, and its ready time is set to 1.164 after calculating the flying time from location 0 to 1. This flying time is obtained by dividing the distance from the depot to location 1 (320.16) by the velocity of drone 2 (275). The parameters for drone 2 ($dr_2 = 1.164$, $dw_2 = 0.44$, $du_2 = 1$, $Route_2 = \{0, 1\}$) and SL ({3, 5, 2, 4, 6}) are updated accordingly. In the second row of Table 3, location 3 is chosen as the next location. AD remains {2, 3, 1} since the unit, weight, and battery capacities of all drones are adequate. Drones 3 and 1 are the most suitable choices with ready times of 0 after excluding *drone 2*. As per the SD list, drone 3 is selected, and the process is repeated.

The subsequent lines follow a similar pattern, with one minor variation in the fifth line of Table 3 where AD becomes {3, 1} due to drone 2's unsuitability based on capacity constraints. The computation of battery capacity requirement for drone 3 on route {0-3-4} is exemplified. Here, the battery capacity is assumed to be linearly related to flying time and payload weight. The total energy requirement is calculated considering the drone's weight, payload weight, flying duration, and a constant (γ). It's crucial to note that the drone must return to the depot after the final delivery.

In this route, drone 3 first flies from the depot to location 3 in 1.846 time units with a total package weight of 0.85 kg (0.60 kg for package 3 and 0.25 kg for package 4). Both packages (package 3 and package 4) are loaded onto drone 3 at the depot. The required battery capacity of drone 3 for flying from the depot to location 3 is computed by multiplying the total weight (the weight of the drone and the weights of the packages) with the flying duration and a constant γ .

$$((6 + 0.85) \cdot (1.846) \cdot 5 \cdot \gamma \cong 63.23\gamma)$$

Similarly, the required battery capacity from location 3 to location 4 and from location 4 back to the depot are respectively computed as 53.625γ and 68.85γ . Suppose that $\gamma = 15$, the total energy requirements are then computed as the sum of these capacities $185.705 \times 15 \approx 2785.58$. Since this total is smaller than the battery capacity of drone 3, the route is feasible.

Once all routes are computed, the service times of locations are calculated by adding the total setup times of customer packages to the related flying time. For instance, the service time of location 1 is computed by summing the flying time (1.164) and the total setup times of packages assigned to the same drone (0.15 + 0.22), resulting in 1.534.

Finally, the lateness of each location is determined by subtracting the service time from the due time. The fitness value of the coot position is then computed by summing the lateness for all customers. This provides an overall measure of how well the drone routing solution meets the timing requirements for package delivery.

Table 3

Computation table for the illustrative example of the solution building algorithm.

SL	<i>i</i>	SD	AD	The Most Appropriate Drones	Selected Drone	Drone Ready Times (<i>dr_i</i>)	Drone Route
{1, 3, 5, 2, 4, 6}	1	{2-3-1}	2-3-1	2-3-1	2	{0} {1.164} {0}	{0} {0-1} {0}
{3, 5, 2, 4, 6}	3	{2-3-1}	2-3-1	3-1	3	{0} {1.164} {1.846}	{0} {0-1} {0-3}
{5, 2, 4, 6}	5	{2-3-1}	2-3-1	1	1	{1.965} {1.819} {1.846}	{0-5} {0-1} {0-3}
{2, 4, 6}	2	{2-3-1}	2-3-1	2	2	{1.965} {2.141} {1.846}	{0-5} {0-1-2} {0-3}
{4, 6}	4	{2-3-1}	3-1	3	3	{1.965} {1.819} {3.562}	{0-5} {0-1-2} {0-3-4}
{6}	6	{2-3-1}	1	1	1	{3.927} {1.819} {3.562}	{0-5-6} {0-1-2} {0-3-4}
Generated routes for drones respectively		: {0-5-6-0} {0-1-2-0} {0-3-4-0}					
Service time of locations respectively		: {1.534, 2.189, 2.206, 3.922, 2.435, 4.397}					
Lateness for locations respectively		: {0, 0.189, 0, 0.922, 0.435, 0.397}					
Total lateness		: 1.943					

5.3. Generating Initial Positions of Coot Population

The COA generally starts the search process with a randomly generated initial population (positions of coots). This random population is repeatedly moved to the different positions with varying objective values in order to find the better solutions (Naruei and Keynia, 2021). In this study, the initial positions of the coots are randomly generated using the well-known formulation expressed in Equation (38) by Naruei and Keynia (2021).

$$CootPos_i^j = rnd(0,1) \cdot (ub - lb) + lb \tag{38}$$

In Eq. (38) $CootPos_i^j$ shows the j^{th} position of coot i , ub and lb represent the upper and lower bounds for a position respectively (in this study $ub=1$ and $lb=0$). In a coot position there are $n + m$ dimensions for the given encoding scheme. Therefore Equation (30) is repeated ($n + m$) times to obtain a complete coot position. Then the fitness value of the coot can be easily determined using the algorithm described in Section 5.2. A population contains a predetermined number of coots (N_{Coot}) and a predetermined number of coot leaders (NL), and they are generated randomly at the initial phase of the algorithm. The fact that the number of coots in the population is the same across all instances is not a rational approach for the handled problem. Therefore, the N_{Coot} is associated with the number of locations (n) in the problem. The number of coots in a population is determined as $N_{Coot} = n \cdot PS(Pop\ Size)$, where the PS coefficient is identified thorough the experimental study. The number of leaders is calculated as $NL=10$ through the preliminary experiments. Once the population is generated, the primary concern is determining the movement of this population.

5.4. Movement Mechanism

Movement of the coots on the surface of the water is modeled by four different mechanisms: random movement, chain movement, position adjustment based on the group leaders, leader-driven movement toward the optimal area. These movements are described in detail in following subsections.

5.4.1. Random movement

Random movement of the coots on the surface is employed to enhance the diversification capability of the algorithm and random movement of a coot is modeled using Eq. (39).

$$CootPos_i^j = CootPos_i^j + rnd(0,1) \times A \times (rnd(0,1) - CootPos_i^j) \tag{39}$$

In the study proposed by Naruei and Keynia (2021) the value of A is computed as $1-L \times (1/Iter)$ where L denotes the number iteration, $Iter$ represents the maximum number of iteration. At the initial phase of the algorithm A is approximately equal to 1, while towards the end of the algorithm it approaches 0. This indicates that the algorithm provides higher diversity in random movement at the beginning, but towards the end, the amount of movement becomes significantly lower. In this study, considering that the stopping condition may differ from the number of iterations, an approach such as $A = (\alpha_1)^L$ was used to

update the A value. α_1 is a continuous number between 0 and 1, and its value was determined through the preliminary experiments.

5.4.2. Chain Movement

Chain movements can be defined as updating the positions of sequential coots in the population based on each other. This movement can be easily performed between two coots. For two coots located next to each other in the population, the position in each dimension is updated by taking the arithmetic average. In the study proposed by Naruei and Keynia (2021) chain movements are executed using the Equ. (40).

$$CootPos_i^j = 0.5 \cdot (CootPos_{(i-1)}^j + CootPos_i^j) \quad (40)$$

5.4.3. Adjusting the position based on the group leaders

In the COA, the groups are led by several coots called group leaders, and the rest of the coots update their positions relative to these leaders. In order for the coots to update their positions relative to the group leaders, it is first necessary to determine which group leader they are assigned to. The following Eq. (41) presented by Naruei and Keynia (2021) is used to determine which group leader the coots are assigned to.

$$k = 1 + (i \bmod NL) \quad (41)$$

where k is the number of leader index, i is the number of coot index and NL is the number of leaders. The position of j^{th} dimension of coot i is updated according to leader k using the Equation (42) presented by Naruei and Keynia (2021).

$$CootPos_i^j = LeaderPos_k^j + 2 \cdot R1^j \cdot \cos(2 \cdot R^j \cdot \pi) \cdot (LeaderPos_k^j - CootPos_i^j) \quad (42)$$

In Eq. (34), $CootPos_i^j$ represents the position of j^{th} dimension of coot i , $LeaderPos_k^j$ shows the position of j^{th} dimension of leader k . $R1^j$ is a random number in the interval $[0,1]$ and R^j is a random number in the interval $[-1, 1]$. Note that these random numbers can be taken from a vector ($R1$ and R) generated randomly at each iteration or they can be randomly generated for each position.

5.4.4. Leading the group leader through the global best coot

The last movement mechanism in the COA is the movement toward the position of the best coot. The group leaders need to update their positions according to the best coot's position. To implement this approach, formula (Eq. (43)) suggested by Naruei and Keynia (2021) is used.

$$LeaderPos_i^j = \begin{cases} B \cdot R3^j \cdot \cos(2 \cdot R^j \cdot \pi) \cdot (gBest^j - LeaderPos_i^j) + gBest^j & \text{rnd}(0,1) < 0.5 \\ B \cdot R3^j \cdot \cos(2 \cdot R^j \cdot \pi) \cdot (gBest^j - LeaderPos_i^j) - gBest^j & \text{rnd}(0,1) \geq 0.5 \end{cases} \quad (43)$$

In Eq. (43) $gBest^j$ shows the j^{th} position of the coot which has the best fitness ever found. $R3^j$ is a random number in the interval $[0, 1]$ and R^j is a random number in the interval $[-1,1]$. As mentioned earlier these numbers can be generated randomly in the algorithm or taken from vectors generated randomly at the beginning of each iteration. B is calculated as $2 \cdot L \times (1/Iter)$ in the study proposed by Naruei and Keynia (2021) where L is the iteration and $Iter$ is the maximum number of iterations. In this study, B is calculated with a formulation $B = 1 + (\alpha_2)^L$ in order to remove the maximum number of iterations from the equation.

5.5. New Movement Approaches for Coot Optimization Algorithm

Classical COA suggested by Naruei and Keynia (2021) contains the approaches given in Sections 6.2 and 6.3. In the classical version, 4 different movement mechanisms are defined and it is aimed to solve optimization problems in line with the standard formulas given for these movements. However, it is possible to increase the effectiveness of the algorithm by using different formulations for movement mechanisms. Therefore, it is desired to investigate the effect of different movement mechanisms on the effectiveness of the algorithm.

In their study, Naruei and Keynia (2021) update the positions of leaders based on their own positions and the best known positions among the coots. However, depending on the problem's structure and characteristics, updating the positions of group leaders may cause them to move to worse positions. Therefore, the performance of searching by preserving the leader positions in the relevant groups for the related problem is also investigated. In classical COA, the leaders in the groups are already updated according to the best positions of the coots. In other words, the best coot position in a coot group is desired to be transferred to the next iteration as a leader. In summary, unlike classical COA, the effectiveness of the algorithm was intended to be observed without using the movement mechanism required to update the group leaders.

$$Leader_k = BestCootPos_k \quad (44)$$

In Eq. (44) while $Leader_k$ represents the position of k^{th} leader, $BestCootPos_k$ shows the best position visited by coots located in group k . In the proposed study, the method that does not apply any movement to the leaders' positions is examined as a different algorithm. This algorithm is abbreviated as COA1 in the following sections of the article.

During chain movement, the new position is typically calculated by taking the arithmetic average of the coots' positions whose fitness values differ from each other. However, it may lead to obtaining a better position by taking the objective function values into consideration when making a chain movement between coots to obtain a better solution. Therefore, a new chain movement mechanism given in Eq. (45) was examined as well in the scope of this study.

$$CootPos_i^j = \frac{(f(CootPos_{i-1}) \cdot CootPos_{(i-1)}^j + f(CootPos_i) \cdot CootPos_{(i)}^j)}{(f(CootPos_{i-1}) + f(CootPos_i))} \quad (45)$$

In Eq. (45) $f(CootPos_i)$ shows the fitness of the position of coot i . In order to increase the weight of better solutions, $f(CootPos_i)$ was calculated as $(1 / Total\ Lateness)$ fitness value in this study. Unlike the classical COA, the algorithm that uses Equation (45) instead of Equation (40) while performing the chain movement is called as COA2. Additionally, in this algorithm, the position of the leaders are not changed and the best coot position of the relevant group is assigned as the leader.

5.6. New Approaches for Selecting the Group Leaders and Regenerating Strategy

In the classical COA, group leaders are continuously updated within certain coot groups. The coot with the best position among the groups to which the coots belong is selected as the leader coot in each iteration, and then a new position is found with the movement mechanism for the leaders. Especially when the number of coots in the groups is high, selecting group leaders by examining not only the coots in their own group but also the coots in the entire population can be more effective in the movement mechanism towards the leader coots. Unlike the classical COA, when updating leaders, first the best NL coot positions are determined for the entire population. The NL number of coot positions obtained in the entire population are assigned as leaders, and the positions of the coots are updated according to these leaders. The important point here is to consider that a coot group may contain more than one leader position.

Since leaders are selected from the entire population, it may happen that the population reaches the best solutions sooner than expected. This situation may cause premature convergence, which is undesirable in heuristic algorithms and negatively affects the solution's quality. In order to prevent premature convergence, the reproduction strategy frequently used in the literature has been added to the COA. According to this strategy, the population is randomly reproduced with a certain probability ($P_{Regenerate}$) in each iteration. When applying the reproduction mechanism, the best coot position is transferred to the next population. The algorithm that takes into account not only the best positions in the group but also the best NL coot position in the entire population when choosing leader positions is called COA3. Again, in this algorithm, the formulation specified in COA2 is used for chain movement and a regenerating strategy is also applied. The general flowchart of the specified COA3 and the COAs can be seen in Fig. 4. In this figure, dashed lines present the general structure of the COA3, while solid lines indicate the general flow of the COA.

Briefly, the effectiveness of four different COAs are examined. The first algorithm includes only the components of the classical COA and no changes are made to the formulation used for the movements of the coots. This algorithm is abbreviated as CCOA in the section where computational experiments are presented. And the other one named as COA1, COA2 and COA3.

5.7. Parameter Tuning for the COA Variants

For the parameter tuning of the COAs, three representative instances (C15_6, C26_7, and C36_9) were selected, as they correspond to the median sizes of the small medium and large sized instances. Each instance was run three times with different parameter values for a duration of $4 \cdot n$ seconds, where n represents the number of nodes in the instance (e.g., 15 nodes and 6 drones for C15_6). The best-performing parameter settings were determined based on the average results from these runs.

The parameters used in the COAs are defined as follows: PS determines the number of coots in each population, proportional to the problem dimension ($N_{Coot} = n \cdot PS$). NL specifies the number of leaders guiding the population. P1 is the probability of using vectors randomly generated at the start of each iteration for R, R1, and R3; if a generated random number exceeds P1, the vectors are generated based on the problem dimension. This parameter influences both diversity and convergence speed. Larger populations enhance exploration but increase computational cost. The parameter α_1 controls the rate of decrease of parameter A for all COAs. α_2 adjusts the decrease of parameter B and is used only in CCOA. $P_{Regenerate}$ defines the probability of regenerating the coot population and is used only in COA3.

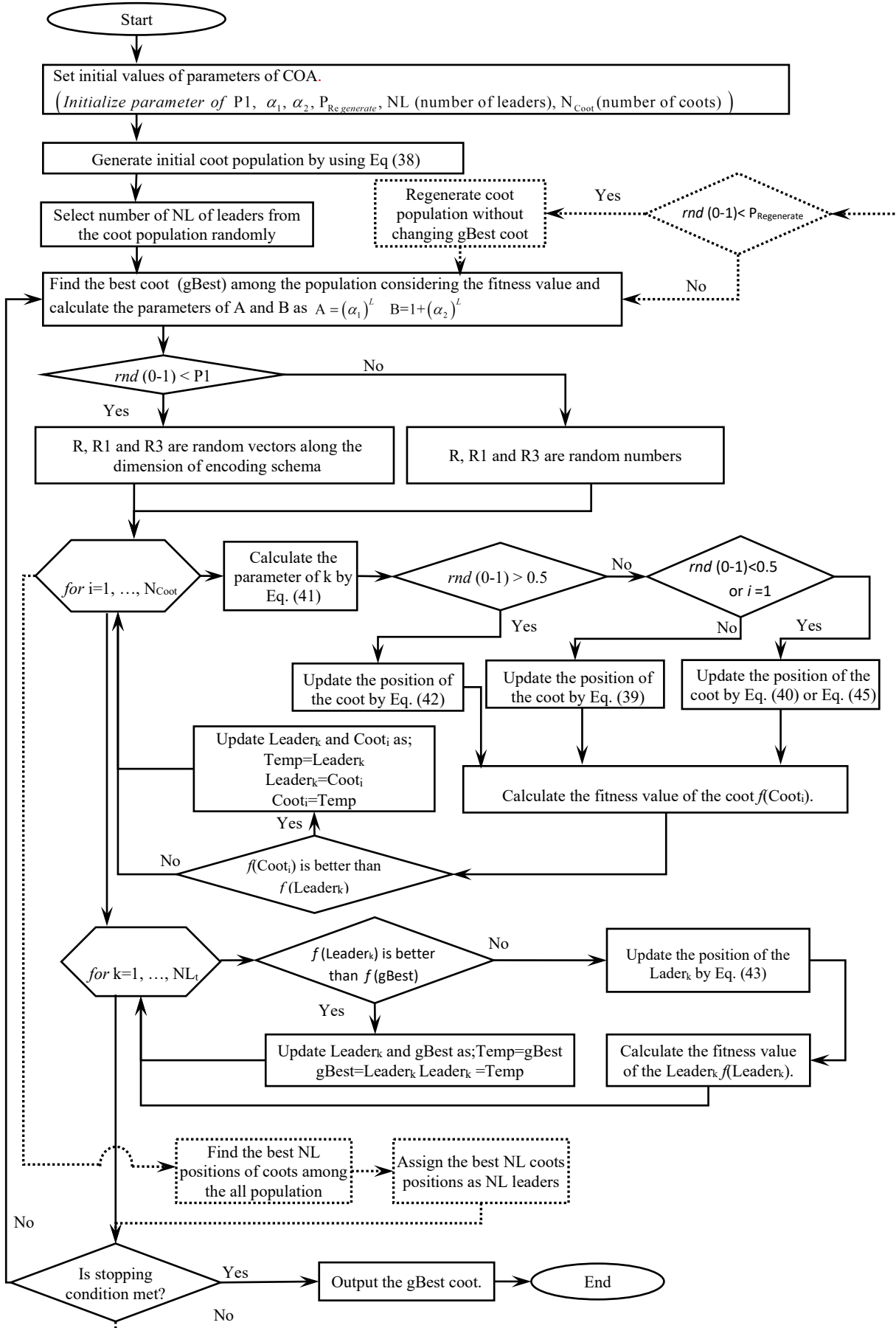


Fig. 4. General structure of the classic and suggested coot optimization algorithm.

The tuning process was performed sequentially rather than exhaustively testing all possible combinations. Initially, extreme values for each parameter were tested across all COA variants. For example, when determining PS, values of 2 and 50 were tested while the other parameters (NL, P1, α , β , $P_{\text{Regenerate}}$) were temporarily set to their respective extreme values to observe their combined effect on performance. PS was gradually decreased from 50 to identify the optimal value, ultimately fixing PS at 8. Similarly, NL was fixed at 10 and P1 at 0.50 for all algorithms. Subsequently, the remaining parameters, including the weight parameters α_1 for CCOA, COA1, COA2 and COA3 (0.80–0.98), α_2 for CCOA (0.80–0.98), and $P_{\text{Regenerate}}$ for COA3 (0.03–0.20), were tuned sequentially. The tested and final selected values for each algorithm are summarized in Table 4.

Table 4
Parameter tuning for the COA variants

Parameter	Tested Values	Selected Values (CCOA)	Selected Values (COA1)	Selected Values (COA2)	Selected Values (COA3)
PS ($N_{\text{Cool}}=n*PS$)	{2, 4, 8, 10, 20, 50}	8	8	8	8
NL	{5, 7, 10, 15}	10	10	10	10
P1	{0.30, 0.40, 0.50, 0.60}	0.50	0.50	0.50	0.50
α_1	{0.80, 0.85, 0.90, 0.95, 0.98}	0.90	0.90	0.90	0.90
α_2	{0.80, 0.85, 0.90, 0.95, 0.98}	0.90	-	-	-
$P_{\text{Regenerate}}$	{0.03, 0.05, 0.10, 0.15, 0.20}	-	-	-	0.05

A PS value of 8, an NL value of 10, and a P1 value of 0.50 were chosen across all COA variants, providing a sufficiently large population for effective exploration, a consistent number of leaders to guide the population, and a balanced probability for using randomly generated vectors, ensuring convergence. $\alpha_1 = 0.90$ was selected to control the decrease of parameter A ensuring a smooth transition from exploration to exploitation. Only COA uses $\alpha_2 = 0.90$ to control parameter B, while COA3 employs $P_{\text{Regenerate}} = 0.05$ to maintain population diversity. These selected values provide a balanced and efficient search process, fully tailored to the problem dimensions and characteristics.

6. Computational Experiments

In this section, the findings obtained from the calculation results of the proposed methods are shared. The computational results of a total of six different solution methods, two mathematical models (Model 1, Model 2) and 4 heuristics (CCOA, COA1, COA2, COA3), are given in this section. Mathematical models were executed for 5400 CPU seconds on computers with an Intel Core I3 processor, running at 2.2 GHZ speed and with 4 GB RAM. Mathematical models were solved by using Cplex 11.2 solver. The COAs are coded in C# programming language and were run five times for $4 \cdot n$ seconds on computers with the same specifications, and the best results were considered.

6.1. Data Set

A total of 63 test instances were created for the heterogeneous DDP, considering the setup time required to load packages into drones. These instances range from the smallest, consisting of 5 cities and 2 drones, to the largest, consisting of 40 cities and 12 drones. As no existing data were found in the literature regarding this specific problem, each test instance was generated independently. The average number of packages per drone in these instances was found to be low, reflecting the limited payload and flight time of drones. Each test sample contains information about the locations of the cities, the latest delivery times of packages to customers, the technical specifications of the drones and the preparation times of customer packages. The location information of the cities is thought to be in two-dimensional space, and the x and y coordinates are created by randomly selecting numbers between 0 and 6000. In each test example, it is assumed that the depot is located at coordinates (1500, 1500). Distances between two cities are determined by calculating the Euclidean distance between them.

6.2. Comparing All Solution Methods

The study compared the performance of six different solution methods on 63 test instances, using the percentage gap (%gap) from the best solution obtained by these methods as the performance criterion (see Eq. (46)).

$$\%gap = 100 \cdot \left[\frac{(f(\text{Sol}) - f(\text{Best}))}{f(\text{Sol})} \right] \quad (46)$$

The results are summarized in Table 5, which categorizes the performance based on the number of nodes in each test instance. As seen in Table 5, there are 3 different test instances containing 5 cities. Model 1 solved all these test instances optimally with an average time of 0.31 seconds. Here, *avr % gap* indicates the average of the deviations from the best results obtained for the relevant test instances.

When the mathematical models are compared in Table 5, it is observed that the two mathematical models are not very different from each other in terms of solving the problem. However, it has been observed that the Model 2 appears to be more advantageous in finding the feasible solutions to the given test instances. It has been observed that, due to the NP-hard structure of the problem, when the problem size exceeds 22 locations, mathematical models cannot even produce an appropriate solution within the given Cpu time limit.

Table 5
Summary of the computational study according to the number of nodes

Problem Info		Model 1		Model 2		CCOA		COA1		COA2		COA3	
#node	#ins.	avr. %	avr.	avr. %	avr.	avr. %	avr.	avr. %	avr.	avr. %	avr.	avr. %	avr.
5	3	0.00	0.31	0.00	2.47	0.00	15	0.00	15	0.00	15	0.00	15
6	3	0.00	3.06	0.00	0.38	0.00	18	0.00	18	0.00	18	0.00	18
7	3	0.00	2.12	0.00	1.04	0.00	21	0.00	21	0.00	21	0.00	21
8	3	0.00	4.47	0.00	5.29	0.00	24	0.00	24	0.00	24	0.00	24
10	3	0.00	78	0.00	11.6	0.00	30	0.00	30	0.00	30	0.00	30
11	3	0.00	324	0.00	44.8	0.00	33	0.00	33	0.00	33	0.00	33
12	3	0.00	34	0.00	877	0.00	36	0.00	36	0.00	36	0.00	36
13	3	0.00*	472	0.00	2804	0.00	39	0.00	39	0.00	39	0.00	39
14	3	0.00*	2688	0.00*	5400	0.61	42	0.32	42	0.56	42	0.00	42
15	3	0.00*	4717	0.00*	5400	0.55	45	0.68	45	0.28	45	0.00	45
16	3	0.00*	3133	0.00	5400	3.33	48	1.86	48	0.60	48	<u>0.26</u>	48
17	3	0.00*	5400	1.54	5400	6.84	51	2.02	51	5.15	51	<u>1.30</u>	51
18	2	2.51	5400	0.00*	5400	2.40	54	2.21	54	<u>1.45</u>	54	2.93	54
20	2	0.00*	5400	0.00	5400	<u>1.20</u>	60	2.26	60	1.58	60	5.10	60
22	2	-	-	-	-	5.26	66	3.21	66	7.35	66	0.61	66
24	3	-	-	-	-	2.82	72	5.24	72	1.98	72	0.77	72
26	3	-	-	-	-	13.22	78	7.31	78	2.72	78	1.30	78
28	3	-	-	-	-	6.35	84	12.91	84	2.79	84	2.00	84
30	3	-	-	-	-	8.39	90	9.52	90	5.77	90	0.00	90
33	3	-	-	-	-	7.38	99	11.37	99	7.42	99	1.81	99
36	3	-	-	-	-	6.35	108	12.91	108	2.79	108	0.17	108
40	3	-	-	-	-	9.66	120	14.58	120	6.61	120	3.94	120

Bold values show the best performances among the given methods.

Underlined values show the best performances among the metaheuristic methods.

(-) :No solution found for the all test instances for the given cpu time limit.

(*) :No solution obtained at least one test problem. *avr. % gap* is computed without them.

(**) :When all instances are considered for the given test problem shows the best performance among the given methods

When the COAs are compared with mathematical models, it can be seen that the COA3 has no difficulty in reaching the solutions obtained by mathematical models up to samples with up to 15 nodes. For the test instances with between 15 and 20 nodes, the COAs produce solutions with acceptable deviations from the known best solutions. When COAs are compared among themselves, COA3 performs the same or better than the other algorithms in 38 out of 40 sample groups. While it was seen that the CCOA showed the best performance in 9 test instances groups, it outperformed COA3 in only 1 sample group. It is also observed that COA2, which uses a different formula in chain movement, achieves better results than the classical method in many sample groups. However, COA2 was found to produce lower-quality solutions compared to COA3.

Table 6
Summary of the computational study according to the number of drones

Problem Info		Model 1		Model 2		CCOA		COA1		COA2		COA3	
#drone	#ins.	avr. % gap	avr. cpu	avr. % gap	avr. cpu	avr. % gap	avr. cpu	avr. % gap	avr. cpu	avr. % gap	avr. cpu	avr. % gap	avr. cpu
2	3	0.00	1.13	0.00	0.47	0.00	18	0.00	18	0.00	18	0.00	18
3	5	0.00	8.65	0.00	4.16	0.00	21.6	0.00	21.6	0.00	21.6	0.00	21.6
4	8	0.00	147	0.00	701	0.00	27	0.00	27	0.00	27	0.00	27
5	9	0.00*	595	0.34	2921	0.40	38.7	0.33	38.7	0.47	38.7	0.00	38.7
6	13	1.43*	2820	0.19*	3856	2.72	54.5	3.81	54.5	2.06	54.5	<u>1.55</u>	54.5
7	11	0.00*	5400	1.13*	5400	5.01	62.7	5.51	62.7	2.17	62.7	2.08	62.7
8	4	-	5400	-	5400	8.11	81	2.96	81	6.83	81	2.15	81
9	3	-	5400	-	5400	11.08	99	10.07	99	5.04	99	5.79	99
10	3	-	5400	-	5400	6.54	109	10.77	109	6.01	109	0.17	109
11	3	-	5400	-	5400	10.18	109	11.94	109	8.98	109	3.94	109
12	1	-	5400	-	5400	11.34	120	15.63	120	10.86	120	0.00	120

Bold values show the best performances among the given methods.

Underlined values show the best performances among the metaheuristic methods.

(-) :No solution found for the all test instances for the given cpu time limit.

(*) :No solution obtained at least one test problem. *avr. % gap* is computed without them.

(**) :When all instances are considered for the given test problem shows the best performance among the given methods

Summaries of the calculation results according to the number of drones are given in Table 6. It can be seen that mathematical models have no difficulty in reaching the optimal solution for test instances with up to 5 drones. It has been observed that mathematical models can achieve feasible solutions in many test examples where the number of drones is between 5 and 7. However, in test cases where the number of drones was 8 or more, it was determined that mathematical models could not produce solutions within the given Cpu time. When COAs and mathematical models are compared, it is seen that the COA3 algorithm can reach the best solutions for the samples containing up to 5 drones. Although mathematical models seem to have better performance compared to COAs in test instances containing 6 and 7 drones, they failed to reach feasible solutions in many of these samples, and these instances were excluded from the *avr % gap* calculation. When the COAs were compared with each other, the COA3 algorithm showed the same or better performance than the other algorithms in 10 of the 11 test sample groups. Compared to COA, COA2 performed the same or better in all sample groups except one.

A summary of the computational experiments based on the quality and status of solutions is presented in Table 7. As seen in Table 7, Model 1 reached the optimal solution in 26 out of 63 examples. However, Model 1 failed to obtain a feasible solution in 31 examples. The average percentage gap of the obtained solutions from the best solutions was calculated as 0.36%. On the other hand, Model 2 achieved 26 optimal solutions but failed to find a feasible solution in 26 test instances. The average percentage gap of the obtained solutions from the best solutions was calculated as 0.22%. It is noteworthy that the best solutions were reached in 32 test instances by Model 2. When heuristic algorithms are compared with the performance of mathematical models on the same table, is evident that they are quite successful in obtaining optimal solutions. It can be seen that COA3 can reach all optimal solutions obtained by the mathematical model. Considering the performance of the heuristic algorithms among themselves, it is seen that COA3 exhibits the best performance in terms of both the number of best solutions obtained and the amount of deviation from the best known solutions.

Table 7

Overview of computational study according to solution status and qualities of solutions.

Method	# of feasible	#of no solution	# of optimal	# of best	<i>avr. % gap</i>
Model1	32	31	26	28	0.36*
Model2	37	26	26	32	0.22*
CCOA	63	0	25**	30	3.57
COA1	63	0	25**	31	3.86
COA2	63	0	25**	37	2.47
COA3	63	0	26**	45	1.31

(*) *avr % gap* for mathematical models are computed considering only test instances on which a feasible solution is obtained.

(**) The optimality of the obtained solution has been proven through a mathematical model.

All these computational experiments indicate that the new formulation used in the chain movement positively enhances the effectiveness of the classical COA for the handled test instances. Additionally, it has been determined that the algorithm obtained by selecting group leaders from the entire population and applying the reproduction strategy significantly improves the performance of the classical COA. In other words, computational experiments demonstrate that the stated algorithm, called COA3, is more effective in multiple aspects (both in solution quality and the number of best solutions) for the heterogeneous drone routing problem with setup times compared to the classical coot optimization method.

7. Conclusions and Future Research Directions

In recent years, advances in battery technology and energy management have enhanced the integration of drones into transportation systems more significant due to their clear advantages over traditional methods. Major companies such as Amazon, FedEx, UPS, Posti, and Google have already begun employing drones for deliveries within specific weight and distance limits. As drone adoption continues to rise, the focus of routing problems can shift from minimizing route lengths based solely on cost to ensuring timely fulfillment of customer expectations.

In this context, this study addresses the drone delivery problem with a focus on reducing total customer waiting times. It considers the diverse capabilities of drones and the time required for loading packages, presenting two mathematical models and four versions of the coot optimization algorithm. Three of these COA variants feature improved mechanisms for movement, leader selection, and regeneration, alongside a classical version. Computational experiments demonstrate that one of the modified COAs outperforms the classical version, providing notable efficiency improvements. Moreover, this study contributes to optimization literature by introducing innovative algorithmic strategies.

Future research could extend this study by exploring several key areas. There's potential for developing models and algorithms that can adapt to dynamic environments, where customer demands and delivery conditions change in real-time. This could involve integrating real-time data and adaptive planning techniques to handle unexpected events like traffic changes or urgent new orders. Additionally, there is potential to focus on specific applications in humanitarian logistics, such as disaster relief operations, to further validate the effectiveness of the proposed algorithms in critical situations. Another avenue for exploration is extending the current single-objective framework to a multi-objective one, simultaneously considering factors

such as cost, energy consumption, environmental impact, and customer waiting times. This holistic approach aims to balance different operational goals and improve overall system efficiency.

Finally, investigating further enhancements to the COA by incorporating machine learning techniques and hybridizing it with other optimization algorithms could lead to improved performance and higher-quality solutions. Addressing these areas in future research would build upon the foundational work presented in this study. Such efforts would promote continued innovation and improvement in the field of drone delivery and optimization algorithms. Ultimately, this could contribute to the development of more efficient, reliable, and sustainable delivery systems capable of meeting the evolving demands of modern logistics.

References

- Abdulsahab, J. A., & Kadhim, D. J. (2022). Robot Path Planning in Unknown Environments with Multi-Objectives Using an Improved COOT Optimization Algorithm. *International Journal of Intelligent Engineering & Systems*, 15(5).
- Akıncılar, A., & Akıncılar, E. (2017). A specific issue on sustainability of transportation planning in an urban region: Ambulance location problem. In *Engineering Tools and Solutions for Sustainable Transportation Planning* (pp. 303-316). IGI Global.
- Aslan, M., & Koç, İ. (2024). Modified Coot bird optimization algorithm for solving community detection problem in social networks. *Neural Computing and Applications*, 36(10), 5595-5619.
- Balasingam, M. (2017). Drones in medicine—the rise of the machines. *International journal of clinical practice*, 71(9), e12989.
- Bruni, M. E., Khodaparasti, S., & Perboli, G. (2023). The drone latency location routing problem under uncertainty. *Transportation Research Part C: Emerging Technologies*, 156, 104322.
- Cheng, C., Adulyasak, Y., & Rousseau, L. M. (2020). Drone routing with energy function: Formulation and exact algorithm. *Transportation Research Part B: Methodological*, 139, 364-387.
- Coelho, B. N., Coelho, V. N., Coelho, I. M., Ochi, L. S., Haghazar, R., Zuidema, D., ... & da Costa, A. R. (2017). A multi-objective green UAV routing problem. *Computers & Operations Research*, 88, 306-315.
- Dorling, K., Heinrichs, J., Messier, G. G., & Magierowski, S. (2016). Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1), 70-85.
- Golmohammadi, A., Bonab, S., & Parishani, A. (2016). A multi-objective location routing problem using imperialist competitive algorithm. *International Journal of Industrial Engineering Computations*, 7(3), 481-488.
- Hong, W., Yin, W., & Xu, S. (2024). Collaborative truck-robot routing problem with meal delivery for the elderly on the personalized needs. *International Journal of Industrial Engineering Computations*, 15(3), 615-626.
- Huang, S. H., Huang, Y. H., Blazquez, C. A., & Chen, C. Y. (2022). Solving the vehicle routing problem with drone for delivery services using an ant colony optimization algorithm. *Advanced Engineering Informatics*, 51, 101536.
- Jazairy, A., Persson, E., Brho, M., von Haartman, R., & Hilletoft, P. (2025). Drones in last-mile delivery: a systematic literature review from a logistics management perspective. *The International Journal of Logistics Management*, 36(7), 1-62.
- Karimunnisa, S., & Pachipala, Y. (2023). Task Classification and Scheduling Using Enhanced Coot Optimization in Cloud Computing. *International Journal of Intelligent Engineering & Systems*, 16(5).
- Kien, L. C., Bich Nga, T. T., Phan, T. M., & Nguyen, T. T. (2022). Coot optimization algorithm for optimal placement of photovoltaic generators in distribution systems considering variation of load and solar radiation. *Mathematical Problems in Engineering*, 2022.
- Kim, H. S., & Kim, Y. (2014). Trajectory optimization for unmanned aerial vehicle formation reconfiguration. *Engineering Optimization*, 46(1), 84-106.
- Kuo, R. J., Lu, S. H., Lai, P. Y., & Mara, S. T. W. (2022). Vehicle routing problem with drones considering time windows. *Expert Systems with Applications*, 191, 116264.
- Liu, Y. (2019). An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. *Computers & Operations Research*, 111, 1-20.
- Lu, Y., Yang, C., & Yang, J. (2022). A multi-objective humanitarian pickup and delivery vehicle routing problem with drones. *Annals of Operations Research*, 319(1), 291-353.
- Luo, Z., Poon, M., Zhang, Z., Liu, Z., & Lim, A. (2021). The multi-visit traveling salesman problem with multi-drones. *Transportation Research Part C: Emerging Technologies*, 128, 103172.
- Meng, S., Guo, X., Li, D., & Liu, G. (2023). The multi-visit drone routing problem for pickup and delivery services. *Transportation Research Part E: Logistics and Transportation Review*, 169, 102990.
- Macrina, G., Pugliese, L. D. P., Guerriero, F., & Laporte, G. (2020). Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies*, 120, 102762.
- Moshref-Javadi, M., & Winkenbach, M. (2021). Applications and Research avenues for drone-based models in logistics: A classification and review. *Expert Systems with Applications*, 177, 114854.
- Moshref-Javadi, M., Hemmati, A., & Winkenbach, M. (2020). A truck and drones model for last-mile delivery: A mathematical model and heuristic approach. *Applied Mathematical Modelling*, 80, 290-318.
- Murray, C. C., & Chu, A. G. (2015). The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54, 86-109.

- Naruei, I., & Keynia, F. (2021). A new optimization method based on COOT bird natural life model. *Expert Systems with Applications*, 183, 115352.
- Özden, A., & İşeri, İ. (2023). COOT optimization algorithm on training artificial neural networks. *Knowledge and Information Systems*, 65(8), 3353-3383.
- Phalapanyakoon, K., & Siripongwutikorn, P. (2021). Route planning of unmanned aerial vehicles under recharging and mission time constraints. *International Journal of Mathematical, Engineering and Management Sciences*, 6(5), 1439.
- Poikonen, S., & Golden, B. (2020). Multi-visit drone routing problem. *Computers & Operations Research*, 113, 104802.
- Raina, A., Nakka, S., Bansal, M., Desai, K. A., Shah, S. V., & Venkatesan, C. (2022). Automated configuration design framework for payload integration in unmanned aerial vehicles. *Engineering Optimization*, 54(12), 2017-2033.
- Sadok, A., Euchí, J., & Siarry, P. (2024). Vehicle routing with multiple UAVs for the last-mile logistics distribution problem: hybrid distributed optimization. *Annals of Operations Research*, 1-41.
- Shavarani, S. M., Nejad, M. G., Rismanchian, F., & Izbirak, G. (2018). Application of hierarchical facility location problem for optimization of a drone delivery system: a case study of Amazon prime air in the city of San Francisco. *The International Journal of Advanced Manufacturing Technology*, 95, 3141-3153.
- Şahin, M. A constructive heuristic for the heterogeneous drone delivery problem that considers packages' setups and battery capacity with the aim of minimizing weighted total waiting times of customers. *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi*, 12(3), 853-862.
- Thibbotuwawa, A., Bocewicz, G., Nielsen, P., & Banaszak, Z. (2020). Unmanned aerial vehicle routing problems: a literature review. *Applied sciences*, 10(13), 4504.
- Torabbeigi, M., Lim, G. J., & Kim, S. J. (2020). Drone delivery scheduling optimization considering payload-induced battery consumption rates. *Journal of Intelligent & Robotic Systems*, 97, 471-487.
- Thomas, T., Srinivas, S., & Rajendran, C. (2023). Collaborative truck multi-drone delivery system considering drone scheduling and en route operations. *Annals of Operations Research*, 1-47.
- Troudi, A., Addouche, S. A., Dellagi, S., & Mhamedi, A. E. (2018). Sizing of the drone delivery fleet considering energy autonomy. *Sustainability*, 10(9), 3344.
- Xu, S., Zong, J., Liu, L., Yang, W., & Xu, L. (2024). An extended PSO algorithm for cold-chain vehicle routing problem with independent loading and minimum fuel volume. *International Journal of Industrial Engineering Computations*, 15(2), 415-426.
- Wen, X., & Wu, G. (2022). Heterogeneous multi-drone routing problem for parcel delivery. *Transportation Research Part C: Emerging Technologies*, 141, 103763.
- Yin, Y., Li, D., Wang, D., Ignatius, J., Cheng, T. C. E., & Wang, S. (2023). A branch-and-price-and-cut algorithm for the truck-based drone delivery routing problem with time windows. *European Journal of Operational Research*, 309(3), 1125-1144.
- Zhang, J., Campbell, J. F., Sweeney II, D. C., & Hupman, A. C. (2021). Energy consumption models for delivery drones: A comparison and assessment. *Transportation Research Part D: Transport and Environment*, 90, 102668.



© 2026 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (<http://creativecommons.org/licenses/by/4.0/>).