Soft Computing
https://doi.org/10.1007/s00500-025-11027-7

OPTIMIZATION

A hybrid airspace model for utilizing drones in logistics operations

Bruno Avelino de Araújo Oliveira[1] · Mauro Caetano[2] · Christopher Shneider Cerqueira[3] · Anderson Ribeiro Correia[2] · Evandro José da Silva[2] · Ronaldo Martins da Costa[4]

Received: 22 November 2024 / Accepted: 13 December 2025

Abstract

This study evaluates the effectiveness of the Biased Random-Key Genetic Algorithm (BRKGA) in solving the Traveling Salesman Problem (TSP) and its extension with trucks and drones, the Flying Sidekick Traveling Salesman Problem (FSTSP). The algorithm was tested on benchmark instances and two real-world scenarios. BRKGA achieved an average gap of 1.4% for TSP and showed better runtime compared to Simulated Annealing (SA) for smaller FSTSP instances. Although it produced faster computational results than SA for medium and large instances, the solution quality was slightly lower. When compared to leading metaheuristics like HTGVNS and HGVNS, BRKGA had competitive runtimes but was less optimal in solutions. In practice, the algorithm reduced delivery times and enabled drones to reach difficult locations. These results suggest that BRKGA is a valuable and effective heuristic for routing problems involving hybrid truck-drone systems.

Highlights

- This study explores unmanned aerial vehicles and their role in the future of air transportation.
- An optimization problem involving two simultaneous vehicles, a truck and a drone, is proposed.
- Using meta-heuristics and mathematical models, two real-world scenarios are examined.
- The optimization algorithm demonstrates time savings with combined truck and drone usage.

Keywords Air transport · Innovation · Optimization · Routing · Truck-drone team logistics

Abbreviations

ACO	Ant Colony Optimization
ADS	Air Domain Study
BRKGA	Biased Random Key Genetic Algorithm
CEI-ITA	Spatial Center ITA
COP	Combinatorial Optimization Problem
CVP-D	Carrier Vehicle Problem with Drones
DDP	Drone Delivery Problem
DO	Drone Operation
DTCO	Drone-Truck Combined Operation

✉ Mauro Caetano
caetano@ita.br

Bruno Avelino de Araújo Oliveira
avel.bruno@gmail.com

Christopher Shneider Cerqueira
chris@ita.br

Anderson Ribeiro Correia
correia@ita.br

Evandro José da Silva
evandro@ita.br

Ronaldo Martins da Costa
ronaldocosta@ufg.br

[1] Postgraduate Program in Operational Applications (PPGAO) / Aeronautics Institute of Technology (ITA) and Brazilian Army (EB), São José dos Campos, SP, Brazil

[2] Brazilian Air Force (FAB) / Aeronautics Institute of Technology (ITA) / Air Transport Laboratory (LabTAR), São José dos Campos, SP, Brazil

[3] Embry-Riddle Aeronautical University (ERAU), Daytona Beach, Florida, US

[4] Informatics Institute, Federal University of Goiás (UFG), Goiânia, GO, Brazil

Published online: 07 February 2026

EIA	Engenharia de Infraestrutura Aeronáutica
FSTSP	Flying Sidekick Traveling Salesman Problem
GA	Genetic Algorithm
GRASP	Greedy Random Adaptive Search Procedure
HGVNS	Hybrid General Variable Neighborhood Search
HTGVNS	Hybrid Tabu General Variable Neighborhood Search
ISR	Intelligence, Surveillance, and Reconnaissance
ITA	Aeronautics Institute of Technology
LB	Lower Bound
mFSTSP	Multiple Flying Sidekick Traveling Salesman Problem
MILP	Mixed Integer Linear Programming
PDSTSP	Parallel Drone Scheduling Traveling Salesman Problem
PPGAO	Programa de Pós-Graduação em Aplicações Operacionais
PSO	Particle Swarm Optimization
RPA	Remotely Pilot Aircraft
RP-D	Routing Problem with Drones
RPV	Remote Pilot Vehicle
RVND	Randomized Variable Neighborhood Descent
TS	Tabu Search
TSP	Traveling Salesman Problem
TSP-D	Traveling Salesman Problem with Drones
UAS	Unmanned Aircraft System
UAV	Uncrewed Aircraft Vehicle
UFG	Universidade Federal de Goiás
UPS	United Parcel Service
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRP-D	Vehicle Routing Problem with Drones

List of symbols

E	Drone Battery Endurance
SL	Time Preparation Before Launch
SR	Time Preparation After Rendezvous
τ^T	Distance Matrix of Truck or Truck
τ^D	Distance Matrix of Truck or Drone

Table 1 Truck and drone complementarity

	Truck	Drone
Speed	low	high
Weight	heavy	light
Energy Consumption	high	low
Capacity	many	unitary
Range	long	short

Source: adapted from Agatz et al. (2018)

1 Introduction

It is expected that the potential market value for drone technology business services will reach several billion dollars and can be regarded as the future of air transportation for various applications in the logistics industry. Drones, including Unmanned Aerial Vehicles (UAVs), Unmanned Aircraft Systems (UAS), Remotely Piloted Vehicles (RPVs), and Remotely Piloted Aircraft (RPAs), expand their operational reach by leveraging the long-distance travel capabilities of trucks, eliminating the need for traditional road infrastructure, avoiding traffic and congestion, and ensuring dependable movement (Moshref-Javadi and Winkenbach 2021; Otto et al. 2018; Rojas Viloria et al. (2021).

In this context, the complementarity between trucks and drones in a multi-modal logistics system stems from their contrasting features. Trucks typically offer lower speeds but have high weight capacity and long ranges, making them suitable for transporting large quantities over great distances with higher energy use. Conversely, drones deliver high speeds and low energy consumption for delivering small quantities over shorter distances, often carrying lightweight loads. This unique set of qualities enables a synergistic strategy where trucks efficiently manage long-distance bulk transport to local hubs. At the same time, drones handle quick last-mile deliveries, utilizing their speed and agility to navigate urban congestion or remote areas. This ultimately aims to improve efficiency and provide greater flexibility, as summarized in Table 1. This combined approach, with a truck acting as a mobile hub for a drone, supports the emerging field of truck-drone team logistics, which explores innovative methods of integrated operations (Akşit et al. 2024; Chung et al. 2020).

Regarding the gap in the literature and this study's contributions, it is noted that although the Flying Sidekick Traveling Salesman Problem (FSTSP) has received significant attention, with various metaheuristics explored for its solution, the application of the Biased Random-Key Genetic Algorithm (BRKGA) in this specific domain remains relatively underexplored. Additionally, existing literature mainly employs compound data structures with separate lists for truck and drone routes, contrasting with the innovative approach introduced here. This research differentiates itself by proposing and implementing a BRKGA framework for the FSTSP, utilizing a unique, simplified data structure where both truck and drone visit points are combined within a single list, with drone-visited locations indicated by negative values. This novel data structure streamlines metaheuristic operations and provides a conceptual advancement in representing hybrid truck-drone routing. Furthermore, to demonstrate the practical applicability and robustness of the proposed method, this study tests the algorithm on

two real-world scenarios: a civilian logistics operation in Florianópolis/Santa Catarina (SC), Brazil, and a military operational scenario, thereby filling a recognized gap in the literature regarding empirical evaluation with real-world data in both contexts.

The problem addressed in this study is the Flying Sidekick Traveling Salesman Problem (FSTSP), a variation of the classic Traveling Salesman Problem adapted for a synchronized truck and drone delivery system. Unlike traditional routing problems that focus on a single vehicle type, the FSTSP involves optimizing the routes of both a truck and one or more drones working together. Key features include serving a set of customers, where each can be visited either by the truck or a drone. The drone has limited battery life and can perform sorties from the truck (which acts as a mobile depot) or a central depot, needing to return within its flight time. The goal is to minimize the total time (makespan) to serve all customers and return both the truck and drone(s) to the depot. This requires optimizing the truck's route, deciding which customers are served by the truck or the drone, selecting launch and rendezvous points (if truck-assisted), and ensuring all operational constraints—such as drone endurance and service times—are met. The main challenge lies in the complex, combinatorial decision of which vehicle serves each customer and the best routing for both, given the synchronization requirements.

These UAV-based benefits are obvious in urban areas, where they shorten delivery times and improve the responsiveness of the logistics system amid ongoing urbanization, rapid growth in direct e-commerce delivery, and increasing congestion. For example, Amazon, Google Wing, United Parcel Service (UPS), and Rakuten have been developing and testing drone delivery models and have recently begun regular commercial drone deliveries, as seen with UPS (Moshref-Javadi and Winkenbach 2021). This innovative logistics solution impacts both economic and environmental factors, drawing interest from customers and logistics operators regarding truck-based drone delivery systems. As a result, existing technology can be integrated with newer solutions, leading to potential incremental innovations in delivery systems, including truck-drone delivery (Baldisseri et al. 2022). This hybrid truck-drone delivery system can be used for various purposes, such as responding to natural disasters with a transportation network; gathering vital information for emergency missions; providing emergency assistance; administering first aid for healthcare; performing intelligence, surveillance, and reconnaissance (ISR) across multiple areas; delivering vaccines; transporting water decontamination tablets and medicines; and resupply deliveries facilitated by drones (González-R et al. 2020).

Understanding the growing field of drone-enabled logistics is crucial because of its potential to transform traditional delivery methods. The combination of increased urbanization, rising city congestion, and the growing challenges of drone use in the military highlights the urgent need for innovative and efficient logistical solutions. Addressing the complexities of last-mile delivery and improving resource use in this changing environment has essential effects on economic efficiency, environmental sustainability, and overall supply chain resilience. Therefore, exploring hybrid truck-drone systems is a timely and critical area of research capable of bringing significant improvements to modern logistics operations.

The literature still needs to improve in implementing better metaheuristic techniques and testing models with real-world military data. This theoretical gap aims to be addressed through a study of applying BRKGA in FSTSP for both civilian and military scenarios. As highlighted by Chung et al. (2020), with the imminent commercial use of trucks and drones, it is crucial to develop a heuristic algorithm that is both efficient and effective.

This study aims to investigate, characterize, and develop route planning optimization for trucks and drones in logistics operations. Specific objectives include implementing and testing the Biased Random Key Genetic Algorithm (BRKGA) within the context of the Traveling Salesman Problem (TSP), implementing and testing BRKGA for the Flying Sidekick Traveling Salesman Problem (FSTSP), and evaluating the model in real-world scenarios using civilian and military data.

2 Solution methods related to routing problems

A Combinatorial Optimization Problem (COP) $P = (S, f)$ can be defined by a search space (or solution space) S with each element s as a candidate solution that satisfies all the constraints and an objective function to be optimized f (Blum and Roli 2003). The solution methods include enumerative procedures and optimization algorithms, such as the simplex algorithm, as well as non-exact strategies like heuristics and metaheuristics. These non-exact approaches can address significant real-world problems, balancing solution quality with reasonable processing time (Peres and Castellari 2021).

Unlike problem-specific heuristics, this methodology is more general and not limited by the specifics of individual optimization problems (Boussaid et al. 2013; Murray and Chu 2015). The word "metaheuristic" comes from the Greek words "heuristic" and "meta," meaning "to find" and "beyond, in an upper level," respectively. It is a non-deterministic algorithm that intelligently explores a search space using various strategies of exploration and exploitation,

employing learning techniques to discover high-quality and near-optimal solutions efficiently (Blum and Roli 2003). The Flying Sidekick Traveling Salesman Problem (FSTSP) is a difficult COP that could benefit from metaheuristics.

Despite the wide variety of metaheuristics, which differ in balancing exploration of the search space (diversification) and exploitation of accumulated search experience (intensification), there is no guarantee that a particular metaheuristic will perform well on all optimization problems, as shown by the hypothesis known as the "No Free Lunch Theorem" (Wolpert and Macready 1997). The macro classification of metaheuristics is based on the number of solutions the algorithm processes at any given moment. If it processes a single solution at a time, it is called single-solution or trajectory-based (example: Greedy Random Adaptive Search Procedure – GRASP). However, if it processes multiple solutions simultaneously, it is classified as population-based (examples: Genetic Algorithm – GA, Ant Colony Optimization - ACO (Boussaïd et al. 2013).

Other classifications shown in Fig. 1 may include memory use for leveraging search history as information for future actions (example: Tabu Search - TS), accepting poorer solutions (example: Simulated Annealing - SA), variable neighborhood structures (example: Variable Neighborhood Search - VNS), large neighborhood structures (example: Large Neighborhood Search), inspiration from Darwin's principle of environmental adaptation (evolutionary computation example: Genetic Algorithm – GA), inspiration from the collective behavior of social insect colonies or other animal groups (swarm intelligence example: ACO), and methods that can combine different metaheuristics and other approaches (Blum et al. 2011; Boussaïd et al. 2013).

Figure 1 shows these different methods encountered in the literature in blue, while the method selected in this study is shown in dark blue. Another metaheuristic, a variant of canonical GA (Holland 1992), with a random-key encoder, mutant insertion, and elitism, is called BRKGA. It demonstrates promising potential, as it has enhanced the performance of GA in the BRKGA proposal and the comparison study by Gonçalves and Resende (2011).

This study uses a precise approach to thoroughly understand the problem and compare its computational results with those in the literature, focusing on performance metrics and solution validation. The BRKGA metaheuristic has also been applied to address the problem across small, medium, and large instances from both the literature and real-world scenarios. The exact and metaheuristic solutions referenced can be identified in Fig. 1 by their dark blue color, while the other approaches shown in light blue help illustrate additional methods from the literature review.

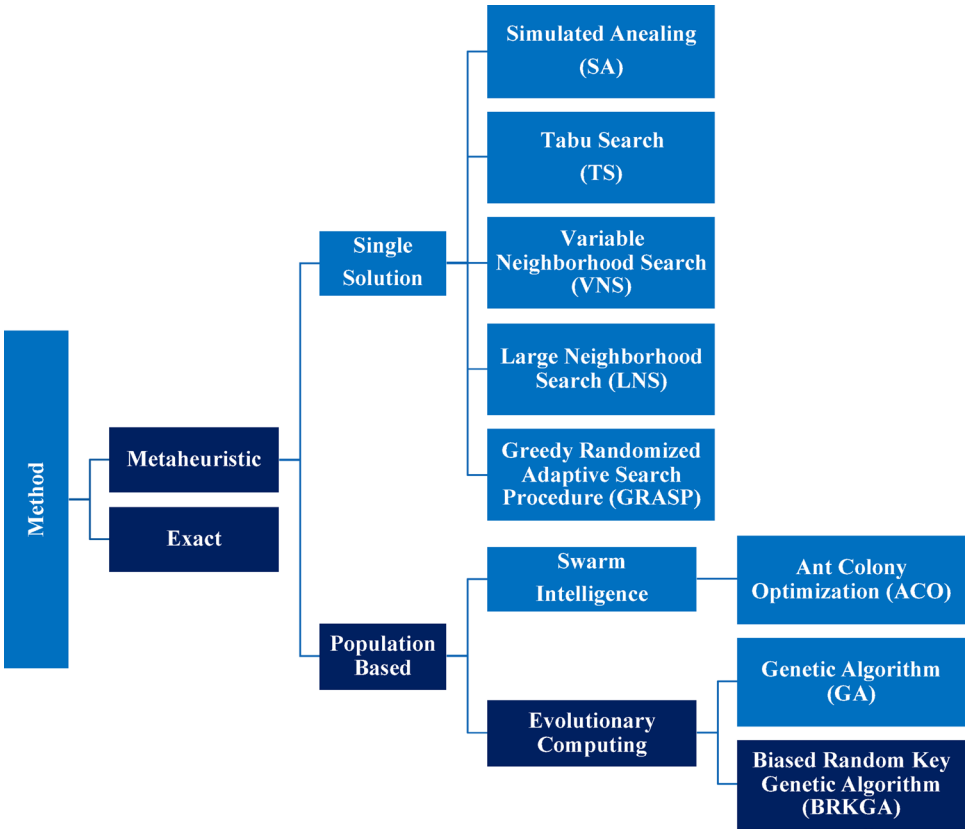

Fig. 1 Solution methods classification.
Source: adapted from Elshaer and Awad (2020).

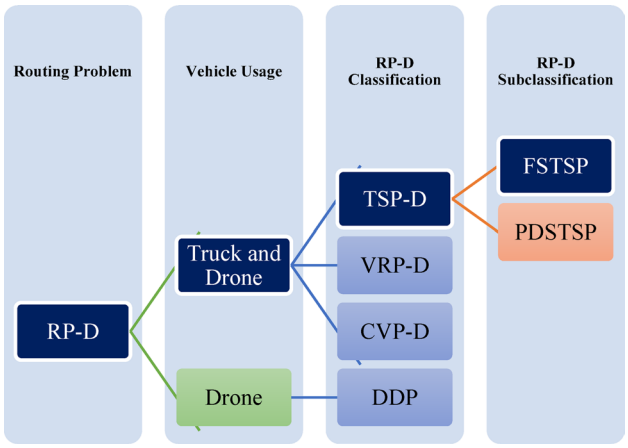

Fig. 2 Routing problem classification.
Source: Adapted from Macrina et al. (2020).

3 Routing problem with drones

There are several variants of drone routing, so it is essential to understand the different usage approaches. In this way, Macrina et al. (2020) classify the Routing Problem with Drones (RP-D) into four categories. The first category involves using drones only for delivery, a problem known as the Drone Delivery Problem (DDP). The second category refers to using one truck and one or more drones as delivery helpers, which is called the Traveling Salesman Problem with Drones (TSP-D). The third approach involves using two or more trucks and drones as delivery helpers, called the Vehicle Routing Problem with Drones (VRP-D). The fourth category is similar to VRP-D, except that only drones make the deliveries. At the same time, trucks are used solely as carriers for one or more drones, known as the Carrier Vehicle Problem with Drones (CVP-D). Figure 2 shows this classification, with RP-D in column 1, vehicle usage in column 2, and the classifications of TSP-D, VRP-D, CVP-D, and DDP in column 3. Besides this classification, other systems exist, like those by Chung et al. (2020), which categorize Drone Operation (DO) as when only a drone performs the delivery, and Drone and Truck Cooperative Operation (DTCO) as when both the truck and drone handle the delivery. However, this study adopts Macrina et al. (2020)'s classification.

In this study, the problem is FSTSP, which falls under the TSP-D classification and is a prime example of truck-drone team logistics. This approach uses one truck and one drone synchronized for delivery, with the truck serving as a mobile base for drone operations. Either the drone or the truck can perform the task, as illustrated in the dark blue boxes in Fig. 2. This choice is based on delivery tasks carried out by ground vehicles like trucks, making it potentially more straightforward to implement drones as auxiliary support to the trucks as an innovative enhancement.

The FSTSP is a variation of TSP where one truck and one drone deliver items in sync, without any loops, as shown in Fig. 3, compared to other routing problems with drones.

In Fig. 3, the routing problem involves a set of customers, each receiving deliveries from either a delivery truck operated by a driver (with the truck's path shown as a solid line and customers served by the truck marked with orange circles) or a drone (with the drone's path indicated by a dashed line and customers served by the drone marked with green circles), coordinated with the truck. The truck must serve some customers on its own because parcels may exceed the drone's payload capacity. Both the truck and drone start from and return to a single depot (represented by a square) exactly once, either together (to save battery while the drone is carried by the truck) or separately. The drone can make multiple sorties, beginning from the depot or the customer location. However, it may need time to change its battery and load parcels before launching. Once launched, the drone must visit a customer and return to the truck or depot within its flight endurance limit. The goal of

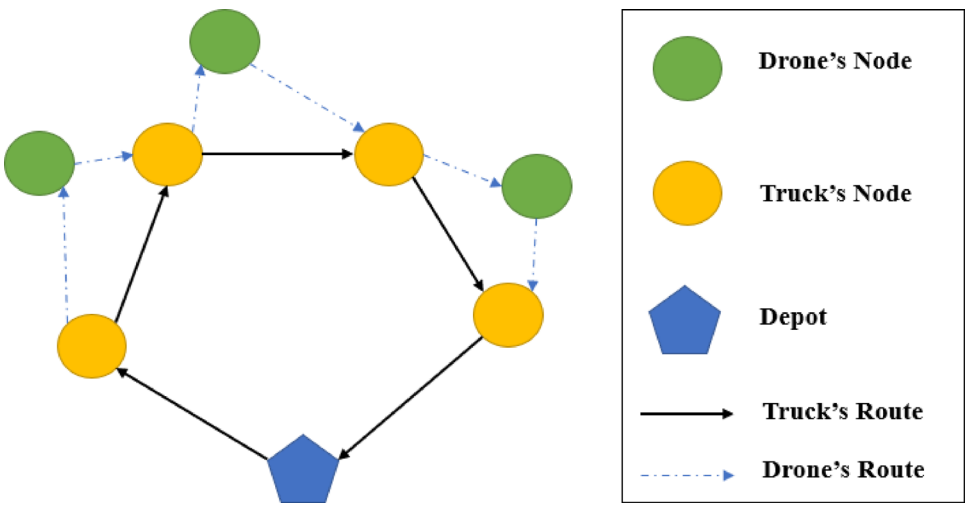

Fig. 3 FSTSP example.
Source: research data.

the FSTSP is to minimize the total time needed to service all customers and return both vehicles to the depot (Murray and Chu 2015).

This variant of the TSP, known as FSTSP, was first introduced by Murray and Chu (2015) as a routing problem that involves both a truck and a drone in serving a set of customers with a single truck synchronized with a drone. The goal is to minimize the total time to serve all customers and to return both vehicles to the depot. Figure 3 shows FSTSP through a delivery operation with one depot containing the products to be delivered, four truck nodes representing customers served by the truck, and three drone nodes representing customers served by the drone, along with five truck paths and six drone paths. Therefore, Fig. 3 illustrates a possible scenario where the truck collaborates with the drone for delivery.

FSTSP is derived from the TSP, a problem where the goal is to determine the route with the lowest cost or distance, ensuring each customer is visited exactly once and the start and end points are the same. Unlike the TSP, FSTSP uses one drone as an auxiliary to the truck vehicles (Murray and Chu 2015).

It is known that, for Macrina et al. (2020), TSP-D is a classification of RP-D. However, according to Agatz et al. (2018), it is a similar problem to FSTSP, another related issue that is nearly the same as FSTSP, except that the drone can wait in the airspace for the truck if it arrives first, enabling loop operations (Agatz et al. 2018). The Parallel Drone Scheduling Traveling Salesman Problem (PDSTSP) is also connected to FSTSP and was first formulated by Murray and Chu (2015). However, in this case, the drone is not synchronized with the vehicle and is recommended for scenarios with many customers within the drone's flight range from the depot as the origin.

As TSP, another classic problem related to FSTSP is the vehicle routing problem (VRP), with the main difference being the number of trucks (two or more). The VRP variant that includes drones is called VRP-D, which features one or more drones assisting the vehicles (Wang and Sheu 2019). Additionally, Fig. 4 illustrates the TSP and VRP variants, including a legend with the depot represented as a blue box, customers as circles, solid arrows as truck routes, and dashed arrows as drone routes, showing traditional TSP and VRP scenarios, as well as variants such as FSTSP, TSP-D, PDSTSP, and VRP-D.

Figure 4 shows the different variants studied. While the primary focus is on the specific variant FSTSP, it's essential to recognize that the literature also covers other aspects of these problems, such as energy efficiency, payload capacity, multiple visits, and dynamic scenarios (Benarbia and Kyamakya 2022; Coutinho et al. 2018; Elshaer and Awad 2020; Khoufi et al. 2019; Macrina et al. 2020; Moshref-Javadi and Winkenbach 2021; Ojeda Rios et al. 2021; Otto et al. 2018; Tan and Yeh 2021). The FSTSP is a variation of the TSP where one truck and one drone make deliveries together, synchronized, without loops, first introduced by Murray and Chu (2015). A similar problem, TSP-D, was first introduced by Agatz et al. (2018), and in this study, it was solved using heuristics based on local search and dynamic programming.

Another issue highlighted in the literature, as noted by Murray and Chu (2015), is the Parallel Drone Scheduling Traveling Salesman Problem (PDSTSP). In this problem, customers can be served either by a single truck or by a fleet of one or more identical drones, to minimize the time when the truck returns to the depot (makespan). There is no

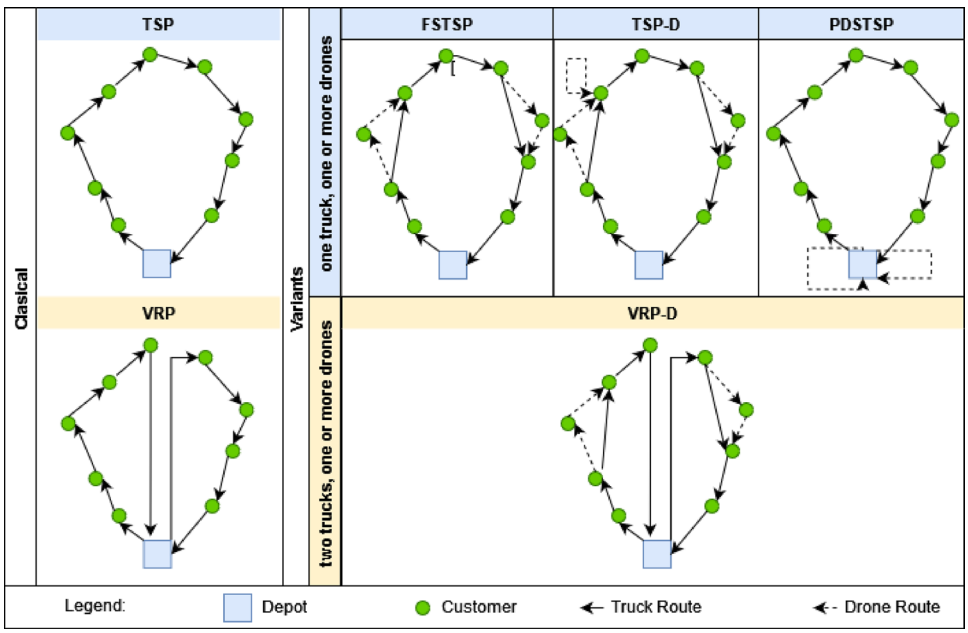

Fig. 4 TSP and VRP variants. Source: Adapted from Macrina et al. (2020).

cooperation or synchronization between the truck and the drones. Additionally, drones start and end their routes at the depot, serve one customer at a time, and can make multiple trips, departing from the depot multiple times. Based on this objective and these constraints, a heuristic is proposed that constructs an initial solution with drones serving all eligible customers and then applies local search to solve PDSTSP.

Several studies propose solutions to Murray and Chu (2015) and Agatz et al. (2018), serving as a basis for various extensions and variants, such as multiple FSTSP (mFSTSP), min-cost TSP-D, and others found in Macrina et al. (2020) and Chung et al. (2020). Additionally, Chung et al. (2020) include TSP-D, FSTSP, and PDSTSP in a class of problems defined as drone and truck cooperative operation (DTCO), where some approaches to solving these problems are discussed in the following paragraphs and summarized in Table 2.

The study by Boccia et al. (2021) addresses an FSTSP to minimize the time required to serve all customers while considering constraints like drone payload capacity and battery life. The new method of representing FSTSP using an extended graph allows for modeling a compact integer linear programming, offering an innovative formulation that avoids the big-M constraints used in synchronized problems. The proposed formulation is solved using an exact branch-and-cut algorithm with a column generation procedure. Tests are conducted with small and medium-sized benchmark instances to demonstrate the competitiveness of this approach.

Another FSTSP algorithm is introduced by Dell'Amico et al. (2021), featuring a branch-and-bound method for small instances with up to 15 customers and a heuristic that employs a branch-and-bound subroutine for larger instances. Experimental results indicate its effectiveness across small, medium, and large instances. Additionally, addressing the FSTSP, De Freitas and Penna (2018) consider an initial solution to the TSP using the Concorde Solver, then apply a heuristic called Randomized Variable Neighborhood Descent (RVND) as a local search to find the solution. They also create 11 instances based on the TSP benchmark to perform computational experiments.

Additionally, De Freitas and Penna (2020) introduce a hybrid heuristic for FSTSP called Hybrid General Variable Neighborhood Search (HGVNS). This method starts by using an exact approach to solve the initial TSP with the Concorde Solver. Then, it applies General Variable Neighborhood Search (GVNS) to improve the delivery routes for trucks and drones. Computational experiments are performed to assess the effectiveness of this solution. More recently, De Freitas et al. (2023) enhanced the best-known solutions in the literature by applying a hybrid metaheuristic named Hybrid Tabu General Variable Neighborhood Structure (HTGVNS).

Also, from a metaheuristic perspective, Ponza (2016) introduces Simulated Annealing (SA) to solve the integer programming formulation by Murray and Chu (2015) for small, medium, and large instances. This study also examines the trade-offs of using faster drones versus drones with longer autonomy through numerical analysis, considering the savings from a combined truck and drone delivery system compared to truck-only delivery.

The related studies include the multiple FSTSP (mFSTSP). Raj and Murray (2020) examine a single truck and a fleet of drones operating at different speeds, offering an innovative approach to drone endurance. This model aims to minimize the total delivery time (or makespan) by treating drone speed as a decision variable, using a three-phased algorithm that dynamically adjusts drone speed to improve performance. The main findings show that operating drones at higher speeds will either reduce or have no negative impact on total delivery time. Additionally, time savings can be achieved by flying drones at variable speeds, and optimizing drone speeds can lead to shorter truck travel distances.

Another related study, the TSP-D proposed by Agatz et al. (2018), is discussed by Roberti and Ruthmair (2020), who introduce a compact mixed-integer linear programming (MILP) model designed to coordinate truck and drone operations effectively. To solve this model, the authors recommend an exact method that includes a branch-and-price algorithm with set partitioning and route relaxation within a three-level hierarchical branching framework, solving instances with up to 39 customers and examining various scenarios.

In contrast, Ha et al. (2018) concentrated on TSP-D to reduce operational costs, including total transportation expenses. They adapted an algorithm from Murray and Chu (2015), which initially produces an optimal TSP solution and then improves it into a feasible TSP-D solution using local search. Additionally, they used a Greedy Randomized Adaptive Search Procedure (GRASP), enhanced with local search and split procedures. They presented numerical results from a variety of instances with different characteristics and sizes in their study.

Building on the evolution of truck-drone route optimization, Yu et al. (2023) addressed the FSTSP using Simulated Annealing (SA) to reduce service time. They described solution encoding, neighborhood structure, and cooling schedule, demonstrating that a well-tuned SA can effectively solve benchmark instances and serve as a firm methodological reference. Expanding on the structural modeling of the problem, Boccia et al. (2024) improved FSTSP modeling by introducing customer classification, which groups deliveries

Table 2 FSTSP literature review

Reference	Model	Objective	Proposed solution	Main contribution
Murray and Chu (2015)	FSTSP, PDSTSP	Min time	MILP, Heuristic	Propose the FSTSP and PDSTSP models for up to 10 customers for a new instance.
Ponza (2016)	FSTSP	Min time	Simulated Annealing	Add two constraints to prevent infeasible solutions and a new set of instances for up to 200 customers.
Agatz et al. (2018)	TSP-D	Min time	Heuristic	Propose a new model that accounts for different speeds (the drone is faster than the truck). A drone can join the truck at the node where it was released (loop). The new instance supports up to 100 customers.
Ha et al. (2018)	TSP-D	Min Cost	Heuristic, GRASP	A new TSP-D variant aims to minimize operational costs with a new instance of up to 100 customers.
De Freitas and Penna (2018)	FSTSP	Min time	RVND	Using drones for last-mile delivery can decrease total delivery time by nearly 20%, reducing the travel distance for trucks.
Raj and Murray (2020)	mFSTSP	Min time	Heuristic	Consider deploying multiple drones instead of just one, as shown by case studies in Buffalo, NY, and Seattle, WA.
De Freitas and Penna (2020)	FSTSP	Min time	HGVNS	A Hybrid General VNS that sets new best-known solutions (BKS) for every FSTSP instance.
Boccia et al. (2021)	FSTSP	Min time	Branch and cut, Column Generation	The proposed branch-and-price method can optimally solve instances with up to 39 customers, outperforming the state-of-the-art.
Roberti and Ruthmair (2020)	TSP-D	Min time	Branch and Price	Advancements in the exact method, with branch-and-cut and column generation, solve instances with up to 20 customers.
Dell'Amico et al. (2021)	FSTSP	Min time	Branch and bound	A branch-and-bound algorithm efficiently handles small instances with up to 15 customers, a heuristic using branch-and-bound manages larger ones, and there have been advances across small, medium, and large instances.
De Freitas et al. (2023)	FSTSP	Min time	HTGVNS	A hybrid Tabu General VNS that created new best-known solutions (BKS) for every FSTSP instance.
Yu et al. (2023)	FSTSP	Min time	MILP Formulation, Simulated Annealing	A well-tuned simulated annealing metaheuristic serves as an efficient methodological benchmark for solving the FSTSP.
Aksit et al. (2024)	FSTSP	Min total arrival Min total tardiness	Mixed-Integer Formulation	A mixed-integer mathematical model that considers energy issues is implemented using the GAMS solver with up to six customers.
Teimoury and Rashid (2024)	FSTSP	Min completion time, operational cost, truck emission, and social penalty	MILP Formulation, HVNS	FSTSP variant with time, sustainability, and rendezvous flexibility, solved using a hybrid metaheuristic (HVNS).
Boccia et al. (2024)	FSTSP	Min time	Machine Learning	FSTSP with customer classification and the introduction of adaptive routing.

Source: research data

based on client categories. Their algorithms use this feature to enhance routing decisions, resulting in a more informed and adaptive optimization approach.

Adding further complexity to the FSTSP, Teimoury and Rashid (2024) expanded the classical formulation by including time dependence, sustainability concerns, and flexible rendezvous points into a new MILP model. To address these real-world factors, they developed a Hybrid Variable Neighborhood Search (HVNS) metaheuristic that combines multiple local search strategies. Their main contribution is modeling a more realistic and time-aware version of the FSTSP while proposing an adaptable metaheuristic capable of managing dynamic logistical constraints.

Building on these efforts with a focus on operational feasibility, Aksit et al. (2024) studied drone energy limits by developing an MILP that combines flight energy consumption to minimize total tardiness and total arrival time. However, it only tackles problems with up to 6 customers. The literature review explores various methods for solving the FSTSP and related issues as proposed by Murray and Chu (2015) and Agatz et al. (2018). Nevertheless, it does not explicitly mention the use of the BRKGA metaheuristic or its implementation with simplified data structures. Additionally, there is a gap in applying these methods to civilian and military scenarios, which presents opportunities for future research to find more efficient and practical solutions.

4 Methodology

This chapter describes the methodology, including a conceptual model, data, and implementation.

4.1 Conceptual model

As the model input, the Ponza (2016) model considers battery endurance (E) as input, which must have a time endurance greater than the drone's travel time from the source i to customer destination j (τ_{ij}^{D}) plus drone travel from customer source j to rendezvous destination k (τ_{jk}^{D}); pre-flight time (SL); post-flight time (SR); truck distance matrix (τ_{ij}^{T}); drone distance matrix (τ_{ij}^{D}). The output includes the truck route and drone route based on the objective functions and related constraints in the model, as shown in Fig. 5.

Next, the problem objectives and constraints involve customer coverage, truck departure and return to the depot, subtour elimination, flux conservation, drone customer launch and rendezvous, coupled movement of trucks and drones, drone and truck time, drone endurance, truck precedence, prohibited movements, initial time, and variable domains. The mathematical model proposed by Ponza (2016) serves as the foundation for understanding the problem and its connections, and guiding the implementation of key elements such as data structures, the objective function, and constraints within this study's framework. The model, described by Eqs. 1–31, provides the basis for understanding the problem and influencing the design of data structures, the objective function, and constraints.

4.2 Mathematical model

Table 3 introduces the notation of the mathematical model, with three columns: the first shows the type, the second displays the mathematical symbols, and the third provides the description.

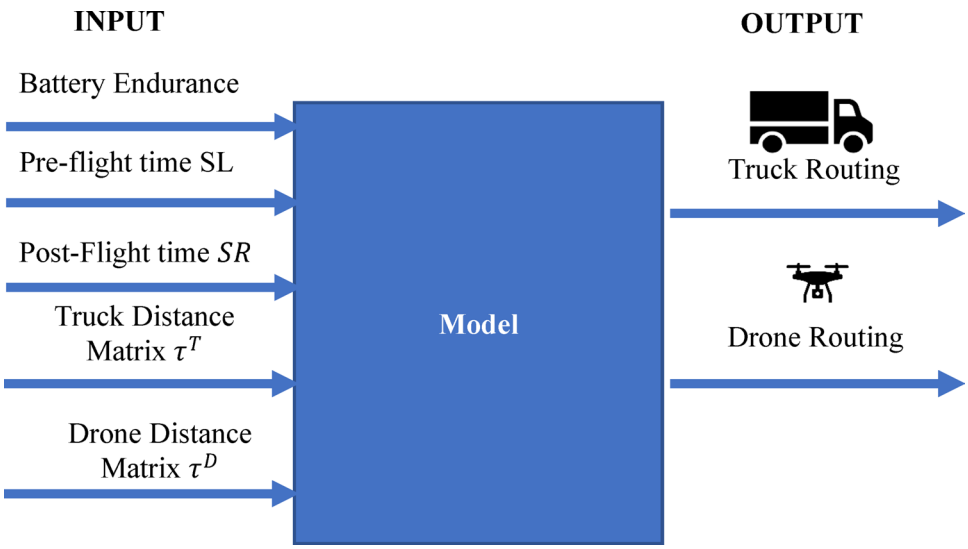

Fig. 5 Model input-output schema. Source: research data.

Table 3 Symbols for variables, paths, sets, and parameters in the mathematical model

Type	Symbol	Description
Decision Variables	x_{ij}	Truck route variable with $x_{ij} = 1$ if the node $j \in N_+$ is visited immediately after node $i \in N_0$, $j \neq i$, and 0 otherwise
	y_{ijk}	Drones sortie representation with y_{ijk}, $\langle i,j,k \rangle \in F$ if the sortie is performed, 0 otherwise
Auxiliary Variables	t_i^D	Time at which the vehicle is prepared to leave from $i \in N$
	t_i^T	Time at which the drone is prepared to leave from $i \in N$
	$p_{i,j}$	Auxiliary binary variable if i is visited by truck before j, with $i \in N_0$ and $j \in C$ with $i \neq j$
	u_i	Auxiliary integer variable to handle with the position of vehicle's path in node $i \in N_+$
Path	**G = (N, A)**	Digraph problem representation with the N as set of nodes and A as set of arcs
	$\langle i, j, k \rangle$	A sortie where $i \in N_0, j \in C\prime$, $k \in N_+$
	(i, j)	Arcs with $i \in N_0$ and $j \in N_+$, $i \neq j$
Sets	**N**	Set {0,1, …, c+1} with all the nodes
	N_0	Set {0,1, …, c} of the nodes representing launching node
	N_+	Set {1, …, c+1} of the nodes representing rendezvous node
	C	Set {1, …, c} of all the customers
	C'	Set $C' \subseteq C$ of the customers available for drone sortie
	A	Set of all arcs (i, j)
Parameters	**F**	Set of all sorties $\langle i, j, k \rangle$ that can be performed within time endurance E
	τ_{ij}^T	Time for truck travel
	τ_{ij}^D	Time for drone travel
	SL	Time for preparing drone's launch
	SR	Time for preparing drone's rendezvous

Source: Ponza (2016)

As the model input, the model of Ponza (2016) considers as input: battery endurance (E), which must have time endurance higher than drone travel from source i to customer destination j (τ_{ij}^D) plus drone travel from customer source j to rendezvous destination k (τ_{jk}^D); pre-flight time (SL); post-flight time (SR); truck distance matrix (τ_{ij}^T); drone distance matrix (τ_{ij}^D). The output includes the truck route and drone route based on the objective functions and relevant constraints in the model.

Next, there are objective problems and constraints related to customer coverage, truck departure and return to the depot, subtour elimination, flux conservation, drone customer launch and rendezvous, coupled movement of trucks and drones, travel times for drones and trucks, drone endurance, prohibited movements for truck precedence, initial timing, and the domain of variables. The goal is to reduce the total time for truck and drone movements, as shown in Eq. 1, which aims to minimize (t_{c+1}^T), that means, the time t of the truck T in the destination ($c+1$).

$$\min t_{c+1}^T \tag{1}$$

Equation 2 guarantees that each city j in set C is visited exactly once during the tour, either by the truck or the assisting drone. The first part of the equation represents the sum of decision variables of truc x_{ij}, the second part represents the sum of drone decision variables y_{ijk}, and the entire expression equal to 1 means that each node (point to be visited) will be visited exactly once, either by the truck or by the drone.

$$\sum_{\substack{i \in N_0 \\ i \neq j}} x_{i,j} + \sum_{\substack{i \in N_0 \\ i \neq j}} \sum_{\substack{k \in N_+ \\ <i,j,k> \in F}} y_{i,j,k} = 1 \; \forall j \in C \tag{2}$$

The truck must depart and return to the depot.

$$\sum_{j \in N_+} x_{0,j} = 1 \tag{3}$$

$$\sum_{i \in N_0} x_{i,c+1} = 1 \tag{4}$$

Equation 5 is a constraint that eliminates subtours through an auxiliary variable decision u.

$$u_i - u_j + 1 \leq (c+2)(1 - x_{i,j}) \quad \forall i \in C, j \in N_+, j \neq i \tag{5}$$

Equation 6 guarantees flux conservation by balancing each flow entering and leaving each customer.

$$\sum_{i \in N_0, i \neq j} x_{i,j} = \sum_{k \in N_+, k \neq j} x_{j,k} \;\; \forall j \in C \tag{6}$$

Each eligible customer can be launched by the drone at most once (Eq. 7) and rendezvous with the drone at most once (Eq. 8).

$$\sum_{\substack{j \in C_d \\ j \neq i}} \sum_{\substack{k \in N_+ \\ <i,j,k> \in F}} y_{i,j,k} \leq 1 \; \forall i \in N_0 \tag{7}$$

$$\sum_{\substack{i \in N_0 \\ i \neq k}} \sum_{\substack{j \in C_d \\ <i,j,k> \in F}} y_{i,j,k} \leq 1 \; \forall k \in N_+ \tag{8}$$

Defines constraints for the movements of both the truck and the drone: if the drone departs to a visit point and then proceeds to a rendezvous point, the truck must travel from the launch point to the rendezvous point (Eq. 9). Equation 10 covers the case where the starting point is the depot.

$$2y_{i,j,k} \leq \sum_{h \in N_+, h \neq i} x_{i,h} + \sum_{l \in N_0, l \neq k} x_{l,k} \;\; \forall i \in N_0, j \in C^d, k \in N_+ \tag{9}$$

$$y_{0,j,k} \leq \sum_{h \in N_0, \; h \neq k} x_{h,k} \; \forall j \in C^d, k \in N^+, <0,j,k> \in F \tag{10}$$

Equations 11 and 12 ensure synchronized launch times for the drone and truck serving customer i by choosing the longer of the two elapsed times. Likewise, Eqs. 13 and 14 handle rendezvous times, ensuring the truck's and drone's coordinated departures or arrivals.

$$t_i^D \geq t_i^T - M \left(1 - \sum_{\substack{j \in C^d \\ j \neq i}} \sum_{\substack{k \in N_+ \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall i \in C \tag{11}$$

$$t_i^T \geq t_i^D - M \left(1 - \sum_{\substack{j \in C^D \\ j \neq i}} \sum_{\substack{k \in N_+ \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall i \in C \tag{12}$$

$$t_k^D \geq t_k^T - M \left(1 - \sum_{\substack{i \in N_0 \\ i \neq k}} \sum_{\substack{j \in C^d \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall k \in N \tag{13}$$

$$t_k^T \geq t_k^D - M \left(1 - \sum_{\substack{i \in N_0 \\ i \neq k}} \sum_{\substack{j \in C^d \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall k \in N \tag{14}$$

Relationship between the duration of a drone's movement.

$$t_j^D \geq t_i^D + \tau_{i,j}^D + SL - M \left(1 - \sum_{\substack{k \in N_+ \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall j \in C^d, i \in N_0, i \neq j \tag{15}$$

$$t_k^D \geq t_j^D + \tau_{j,k}^D + SR - M \left(1 - \sum_{\substack{i \in N_0 \\ <i,j,k> \in F}} y_{i,j,k} \right) \forall j \in C^d, k \in N_+, k \neq j \tag{16}$$

The correlation between truck movement completion times.

$$\begin{aligned} t_k^T \geq t_h^T + \tau_{h,k}^T + SL & \left(\sum_{\substack{l \in C^D \\ l \neq h}} \sum_{\substack{m \in N_+ \\ <h,l,m> \in F}} y_{h,l,m} \right) \\ + SR & \left(\sum_{\substack{i \in N_0 \\ i \neq k}} \sum_{\substack{j \in C^D \\ <i,j,k> \in F}} y_{i,j,k} \right) \\ - M & (1 - x_{h,k}) \; \forall h \in N_0, k \in N_+, k \neq h \end{aligned} \tag{17}$$

Limits the total flight time of the drone movement according to battery endurance E.

$$t_k^D - t_i^D \leq E + M \left(1 - \sum_{j \in C^D, <i,j,k> \in F} y_{i,j,k} \right) \forall k \in N_+, i \in N_0 \tag{18}$$

Equations 19, 20, and 21 define precedence constraints between customer visits by the truck.

$$u_i - u_j \geq 1 - (c+2)\, p_{ij} \;\; \forall i \in C, j \in C, j \neq i \tag{19}$$

$$u_i - u_j \leq -1 + (c+2)\,(1 - p_{ij}) \; \forall i \in C, j \in C, j \neq i \tag{20}$$

$$p_{ij} + p_{ji} = 1 \tag{21}$$

Equation 22 enforces a constraint that prevents unsafe situations in solving the problem, such as when a drone is launched before completing a rendezvous or when a drone has both a launch and a rendezvous occurring after a launch.

$$t_l^D \geq t_k^D - M\left(3 - \sum_{\substack{j \in C^D, j \neq k \\ <i,j,k> \in F}} y_{i,j,k} - \sum_{\substack{m \in C^D \\ m \neq i,\ m \neq k, m \neq l}} \sum_{\substack{n \in N_+, <i,j,k> \in F \\ n \neq i, n \neq k}} y_{l,m,n} - p_{i,l}\right) \quad \forall i \in N_0, k \in N_+, l \in C, k \neq i, l \neq i, l \neq k \tag{22}$$

Equations 23 and 24 specify the initial movement times for the truck and drone, respectively. Equation 25 defines the initial auxiliary decision variable p.

$$t_0^T = 0 \tag{23}$$

$$t_0^D = 0 \tag{24}$$

$$p_0, j = 1 \ \forall j \in C \tag{25}$$

Define the domain for the variables $x_{i,j}, y_{i,j,k}, u_i, t_i^T, t_i^D, p_{i,j}$

$$x_{i,j} \in \{0,1\} \ \forall i \in N_0, j \in N_+, j \neq i \tag{26}$$

$$y_{i,j,k} \in \{0,1\} \ \forall i \in N_0, j \in C^D, k \in N_+, j \neq i, <i,j,k> \in F \tag{27}$$

$$1 \leq u_i \leq (c+2) \forall i \in N_+ \tag{28}$$

$$t_i^T \geq 0 \forall i \in N \tag{29}$$

$$t_i^D \geq 0 \forall i \in N \tag{30}$$

$$p_{ij} \in \{0,1\} \ \forall i \in N_0, j \in C, j \neq i \tag{31}$$

.

Table 4 Parameters and features of Ponza dataset (2016)

Category	Value
Average Speed	Truck: 35 mph (56 km/h or 15 m/s) Drone: 50 mph (80 km/h or 22 m/s)
Service Times	Before launch (σ^L): 40 s After rendezvous (σ^R): 30 s
Drone Battery Endurance	1440 s (24 min)
Correction Factor ($\boldsymbol{\nu}$)	0.8 (accounts for acceleration, deceleration, traffic lights, etc.)
Travel Time Matrix ($\boldsymbol{\tau}_{ij}^{\boldsymbol{T/D}}$)	$\tau_{ij}^T = \frac{d_{ij}}{(truck\ speed \times \nu)}$ $\tau_{ij}^D = \frac{d_{ij}}{(drone\ speed \times \nu)}$
Drone-Serviceable Customers ($\boldsymbol{cDrones}$)	80% of customers, randomly selected
Ineligibility Reasons	Geographical limitations, heavy parcels, etc.
Dataset Instances	Total of 50 instances: - Small: 5, 6, 7, 8, 9, 10 customers - Medium: 50, 100 customers - Large: 150, 200 customers

Source: research data

The mathematical model proposed by Ponza (2016) and described by Eqs. 1–30 provides the foundation for understanding the problem and its connections, and it guides the implementation of key elements such as data structures, the objective function, and constraints within this study's framework.

4.3 Test instances and data

Since Ponza's (2016) data serves as the benchmark model, this study uses the same parameters. The truck's average speed is set at 35 mph (56 km/h or 15 m/s), while the drone's average speed is 50 mph (80 km/h or 22 m/s). The service time before launch is 40 s, and after rendezvous, it is 30 s. The drone's battery endurance lasts 1440 s (24 min). The truck distance matrix reflects a speed of 35 mph, and the drone distance matrix considers a speed of 50 mph.

Artificial data from existing literature is essential for benchmarking the algorithm against others. Therefore, the dataset from Ponza (2016) has been used, which considers the number of customers, average truck speed, average drone speed, and service time before launch (σ^L), service time before rendezvous (σ^R), flight time endurance of the drone (E), big enough number to be used in mathematical equations strategies (M), set of all eligible customers serviceable (cDrones), set of all drone's possible sorties (F), distance matrix considering Euclidian distances (d_{ij}), truck travel time matrix (τ_{ij}^T), drone travel time matrix (τ_{ij}^D). These τ_{ij}^T and τ_{ij}^D is calculated according to the formula $\tau_{i,j}^{T,D} = \mathrm{d}_{i,j}/(\mathrm{speed}^{\mathrm{T,D}} * \nu)$, with $speed^T$ as the truck constant average speed of 35 mph, and $speed^D$ as the drone constant average speed of 50 mph, ν is a generic coefficient of acceleration, deacelerations, traffic lights, road network, assumed to be $\nu = 0.8$. The set of serviceable customers for the drones (cDrones) is selected randomly, with 80% of customers being serviceable. This selection is made without a specific reason, but in a real-world scenario, it can be due to geographical limitations, too heavy parcels, and other real-world criteria (Ponza 2016).

The dataset includes 50 instances, each with a different number of customers. These instances feature 5, 6, 7, 8, 9, and 10 customers, as well as medium and large cases with 50, 100, 150, and 200 customers. Smaller instances (ranging

from 5 to 10 customers) are suitable for exact methods and optimal solutions. In contrast, medium-sized instances (ranging from 50 to 100 customers) are better suited for heuristic methods, offering insights into real-world scenarios. Larger instances with 150 and 200 customers are used to test the model's capabilities extensively.

Each of these instances reflects the idea that Amazon and HorseFly drones have a 10-mile endurance in a drone sortie (Ponza 2016). The x and y node coordinates are randomly generated numbers, as shown in Table 4, except for the fixed depot, which is always at (0,0).

The literature indicates that few studies utilize real-world data in the FSTSP. Therefore, this paper presents a real-world scenario using the coordinates of a civilian and a military logistics operation.

To evaluate the FSTSP model with real-world data, this study presents two scenarios: civilian and military operational data. The Haversine distance is used to assess the distance matrix, as demonstrated by the nodes' locations in Fig. 6, rather than the Euclidean distance employed in Ponza (2016). The reason for this choice is that the Haversine distance considers the Earth's curvature, offering a more accurate measure of geographic distances in real-world situations. The velocity parameters for both the truck and drone remain consistent with those defined by Ponza (2016), with a standardized acceleration coefficient (v) set to 0.8. Additionally, following Ponza's methodology, this study uses the same values for SL (Service Launch) and SR (Service Rendezvous), which are 40 s and 30 s, respectively. It also maintains an endurance threshold (E) of 1440 s (equivalent to 12 min).

Two separate case studies were conducted: the first in Florianópolis, Brazil, covering nine locations, including deliveries to the town center, the beach, and the oil platform at Florianópolis Air Base. This scenario is also used by the Spatial Center ITA (CEI-ITA) as part of the Air Domain Study (ADS) Brazil-Sweden Project, providing a relevant context for evaluating performance in civilian urban delivery operations. The second case involved a military operation scenario with 10 points, including a maintenance point, an observation point, an operational base, and distribution points for class III and V supplies and airbases. It is based on a real maneuver that can be used in a delivery simulation. For security reasons, Fig. 7 has been modified to prevent the identification of military locations.

The first case study in Fig. 6 examines a civilian urban delivery scenario in Florianópolis, Brazil. This scenario includes a network of nine different locations, carefully selected to reflect the diverse geographic and logistical challenges of a modern city. These locations feature deliveries in the busy town center, coastal deliveries on the beach, and more complex destinations such as the oil platform at Florianópolis Air Base. This case study is especially relevant because it aligns with the Spatial Center ITA's (CEI-ITA) Air Domain Study (ADS) Brazil-Sweden Project, highlighting its significance within the academic and research community for evaluating the performance of innovative delivery solutions in real-world civilian settings, particularly in urban delivery operations.

The second case study illustrates a military operation scenario shown in Fig. 7, featuring a network of ten distinct points, each representing a critical element of a real

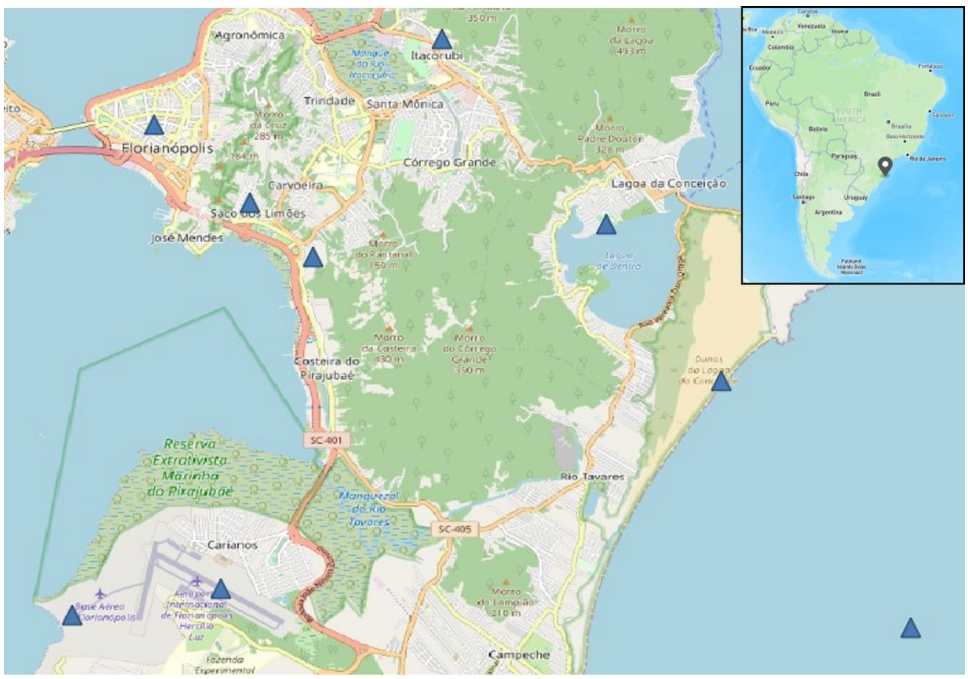

Fig. 6 Civilian scenario. Source: research data.

Fig. 7 Military operation. Source: research data.

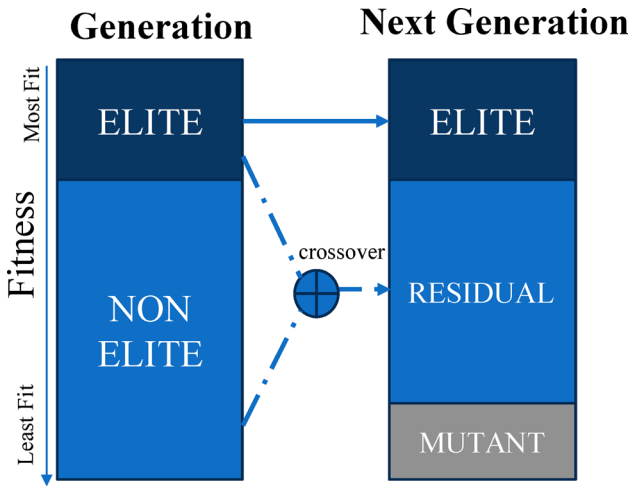

Fig. 8 Elitism Procedure in BRKGA. Source: research data.

maneuver adapted for delivery simulation. These points include a maintenance point for equipment servicing, an observation point for surveillance, an operational base serving as the central command hub, and a distribution point for class III supplies (typically fuel and lubricants) as well as class V supplies (ammunition and explosives), among various airbases. Due to military sensitivity and data security protocols, the geographic locations in Fig. 7 have been intentionally altered to prevent the identification of specific military installations while maintaining the operational complexities and logistical challenges inherent in such scenarios. This case study provides a valuable opportunity to evaluate the FSTSP model's applicability and effectiveness in a demanding military operational setting.

4.4 Model implementation using Biased Random Key Genetic Algorithm (BRKGA)

The BRKGA is a metaheuristic algorithm used in optimization problems, incorporating several key concepts:

- Genetic Algorithm (GA): BRKGA is a specialized type of GA inspired by natural selection principles. GA evolves a population of potential solutions over multiple generations to improve quality.
- Random-Key Encoding: It uses random-key encoding to represent potential solutions, employing real numbers within the [0, 1] range as keys to construct solutions.
- Biased Sampling: This method introduces bias into the sampling process by favoring certain regions of the solution space over others. Essentially, BRKGA uses these ideas to tackle optimization problems, making it especially effective in scenarios with complex solution spaces and the need for efficient exploration.

Figure 8 visually shows how BRKGA develops over several generations to improve solution quality. Generations are organized by ranking solutions based on fitness, emphasizing both elite and non-elite solutions. In each iteration, one elite solution and one non-elite solution are randomly chosen for the crossover process. The elite population is directly carried over to the next generation, and a set of randomly generated mutant solutions is added. This ongoing process leads to continuous improvement and refinement of solutions from one generation to the next.

In this evolutionary process shown in Fig. 8, the population is divided into an elite group, which includes the best solutions in the generation, and a non-elite group made up of the rest. Then, an individual is assigned to either the elite or non-elite group for a crossover operation, which is biased

since an elite solution is always chosen. The elite group is carried over to the next population according to the p below the line i, and this parameter is used to preserve and improve good solutions. To generate new solutions, a mutant population is created based on the p below the line m and the rate, forming the next generation.

4.4.1 Data structure

The data structure used in the BRKGA implementation consists of a list of points to be visited by a truck or drone. When the number is positive, the truck visits, and when the number is negative, the drone visits. Compared to other studies in the literature, such as those by Murray and Chu (2015) and Ponza (2016), this study integrates trucks and drones within the same data structure, a conceptual advancement not found in the literature except for a preprint by Mahmoudinazlou and Kwon (2023). Consequently, it lays the groundwork for more efficient heuristic algorithms in truck and drone routing optimization, as recognized by researchers like Chung et al. (2020). This approach can reduce complexity by performing metaheuristic operations using just one data structure. It explores the application of a previously unused metaheuristic to this problem, introduces new methods for route optimization with drone assistance, and offers new insights by implementing a BRKGA encoder that incorporates real numbers.

BRKGA uses a unique data structure with a single list that includes both truck and drone components. It distinguishes between the two using negative numbers designated solely for drone flights. In contrast, other studies on routing problems with drones employ different data structures, often based on a combined approach with separate lists for trucks and drones (Agatz et al. 2018; De Freitas and Penna 2018; Freitas and Penna 2020; Ha et al. 2018; Murray and Chu 2015; Ponza 2016; Raj and Murray 2020).

As shown in Fig. 9, the route is visually represented and documented with data in the list format [1, 1, 1,2, −3, 4, 3, 4, 3,4,5], along with the truck path, drone path, depot, and the nodes for both truck and drone, as indicated in the legend of Fig. 9. The truck moves from the central point 0 to points 1, 2, 4, 5, and then back to 0. Meanwhile, the drone is mounted on the truck during arcs 0–1, 1–2, 4–5, and 5−0, but not during arc 2–4, as it is visiting point 3. The drone departs from point 2 with a pre-flight time (SL), visits point 3, flies to point 4 with a post-flight time (SR), and continues traveling with the truck. To compute the objective function value of the tour, an iterative process must be used to calculate each arc of the truck and drone paths.

The data representation in Fig. 9 is for convenience, without a starting point of 0, to facilitate algorithm execution. However, the truck always starts and arrives at this point. As shown in Fig. 9, the truck travels from 0 to 1, 1–2, 2–4, 4–5, and 5−0, but not from 2 to 3 and 3–4 because these segments represent the drone's path. In this section, two

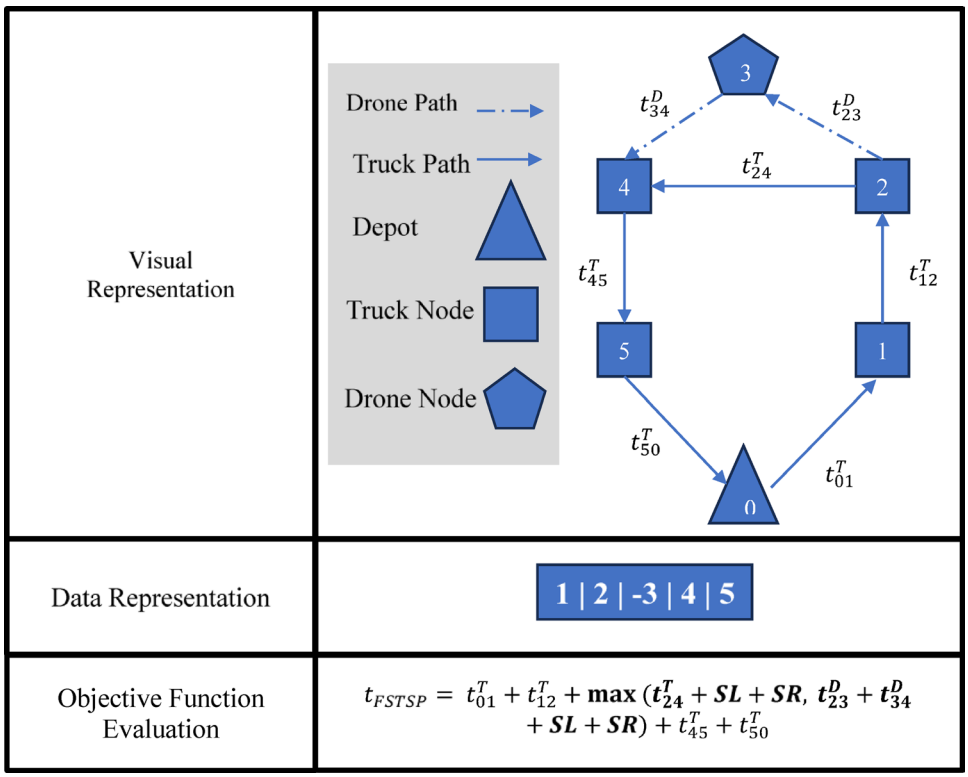

Fig. 9 Visual representation of FSTSP. Source: research data

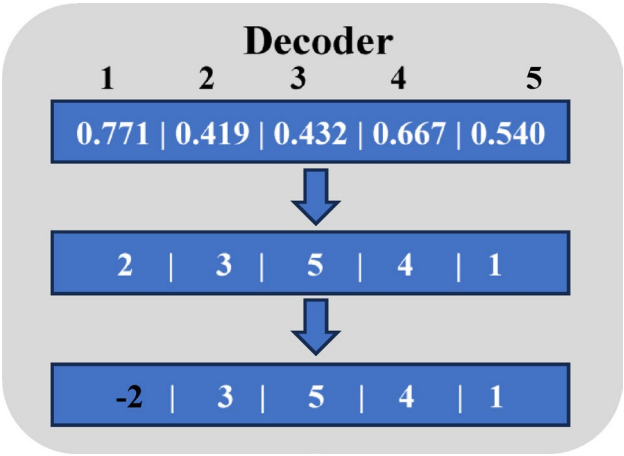

Fig. 10 BRKGA decoder with type-aware chromosome. Source: research data

scenarios may occur: first, the truck arrives at point 4 and waits for the drone; second, the drone arrives at point 4 and waits to fly to the truck, depending on battery endurance. These two scenarios are characterized by the maximum time between them, as the truck and drone are synchronized. In the BRKGA algorithm, a penalty is added to the cost function when the battery endurance condition is violated.

4.4.2 Algorithm initialization and solution representation

The algorithm starts by creating an initial set of potential delivery schedules. It randomly arranges these schedules and immediately considers options for both trucks and drones. It determines the sequence of customer visits and assesses the cost-effectiveness of using either a truck or a drone for each delivery, including the time needed for drone launch and return. This initial evaluation helps generate a preliminary feasible solution that uses both transportation modes. The algorithm then explores variations around the best solutions found so far to identify neighborhood solutions. New schedules are generated by making slight adjustments based on these promising solutions, combining elements from them, and adding some randomness to expand the search. Each new possible solution is evaluated based on the costs of trucks and drones, enabling the algorithm to gradually improve the delivery plan and select the best transport mode for different parts of the route.

The underlying structure, known as the chromosome structure, represents a potential delivery plan and encodes the sequence in which customers might be visited. Within this structure, the algorithm implicitly decides whether a delivery to a specific customer should be made by truck or drone. This decision is based on calculating and comparing the costs for each option, effectively 'assigning' each delivery task to the most suitable mode to minimize costs or time. Notably, a point visited by a drone is usually marked as negative to distinguish it from other locations in the delivery sequence.

4.4.3 Decoder

he BRKGA decoder is a key step in the BRKGA algorithm because it can result in high computational costs and significantly affect the solution quality. As shown in Fig. 10, the chromosome is created with a length equal to the number of points to visit, represented by chromosome elements, also known as genes. Each gene in BRKGA is expressed as a random key, a real number between 0 and 1.

In the example from Fig. 10, a chromosome is represented as a list: [0.771, 0.419, 0.432, 0.667, 0.540], with indices arranged in ascending order by gene value: [2, 3,

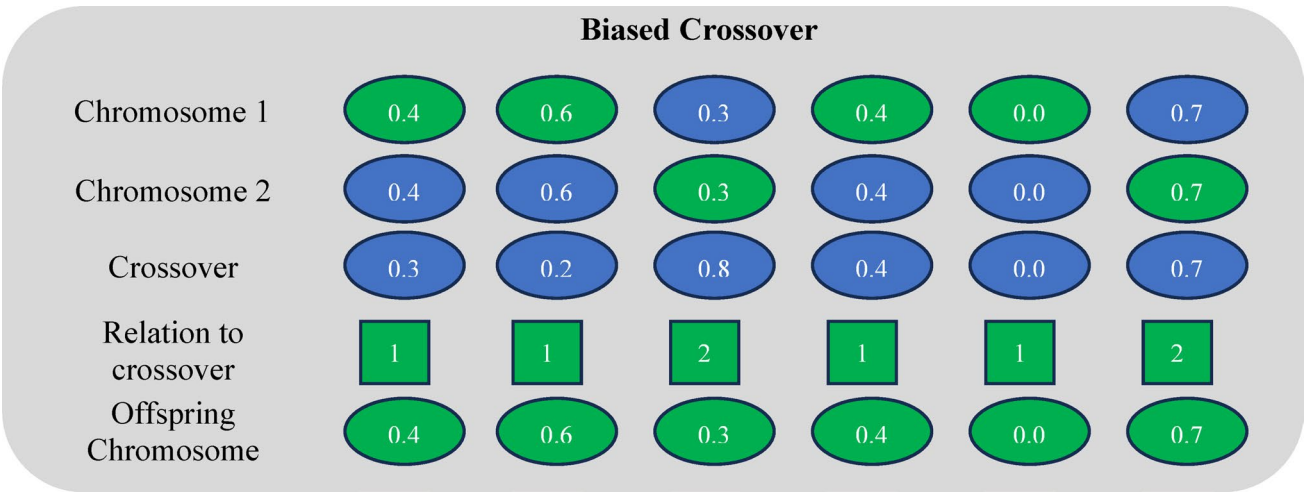

Fig. 11 Biased crossover. Source: research data

5, 4, 2,3,5,4,1]. The first gene is made negative because the drone costs less than the truck and has enough battery and condition to complete the task. This indicates a drone sortie visiting point 2 from point 0 to point 3.

4.4.4 Biased crossover

These BRKGA crossover operators, known as biased crossover, function as elitist selection and population combination, because the best solutions are more likely to be chosen for the elite group. In Fig. 11, a chromosome from the elite group, chromosome 1, is combined with a non-elite chromosome, chromosome 2. Based on the rate of 0.7, only genes 0.35 and 0.78 are selected from chromosome 2, resulting in the offspring chromosome [0.45, 0.67, 0.35, 0.49, 0.07, 0.78].

4.4.5 BRKGA flowchart

The BRKGA algorithm was chosen for this study because of its strong performance in other optimization problems and its ability to prioritize and select genes based on specific criteria for drone sorties.

Pseudocode Algorithm 1 outlines the steps from initialization to completion, as shown in the pseudocode. It starts by setting problem parameter (drone cost τ^D – the drone cost of movement from one to another point, truck cost τ^T - the truck cost of movement from one to another point, SL – pre-flight time, SR – post-flight time, E – battery endurance, V_{drone} – set of points that drone can flight) and algorithm parameter (p_{elite} – elite probability, p_{mutant} – mutant rate, $n_{generations}$ – number of generations, $not_{improve-stop}$ – number of runs without improvement; P – initial population, it means, the number of individuals (solutions) in each generation). It has been used the next parameters to this algorithm implementation: p_{elite} of 20%; p_{mutant} of 10%; $n_{generations}$ of 500; P of 500; $not_{improve-stop}$ of 15. The choice of these parameters was based on both existing applications of the BRKGA algorithm in the literature and empirical testing. In the future, these parameters can be further refined to attain optimal computational results.

The process starts by creating an initial population P and identifying the best individual. Then, an iteration begins, continuing until reaching the maximum number of generations (lines 2–11) or the limit of runs without improvement (lines 8–10). Within the population, individuals are decoded, and their fitness is computed (line 3) to rank them based on fitness and classify them as either elite or non-elite populations (line 4). A mutant population is generated (line 5) to introduce new individuals, and a biased crossover is performed using parametric uniform crossover (PUX) between elite and non-elite groups, as shown in line 6. Line 7 shows the union of the elite, mutant, and offspring populations, while lines 8–10 indicate that the algorithm stops if there is no improvement in solutions.

Similar to a flowchart that clarifies the information flow in the algorithm, the BRKGA flowchart in Fig. 12 illustrates the process, starting with generating a random key and decoding each list of random keys. Then, the iteration continues until the stopping rule criterion is met. The population is ordered by fitness and divided into elite and non-elite groups; the next generation is created by copying the elite to the new population, generating mutants, and performing biased crossovers between elite and non-elite individuals until the stopping rule halts the process.

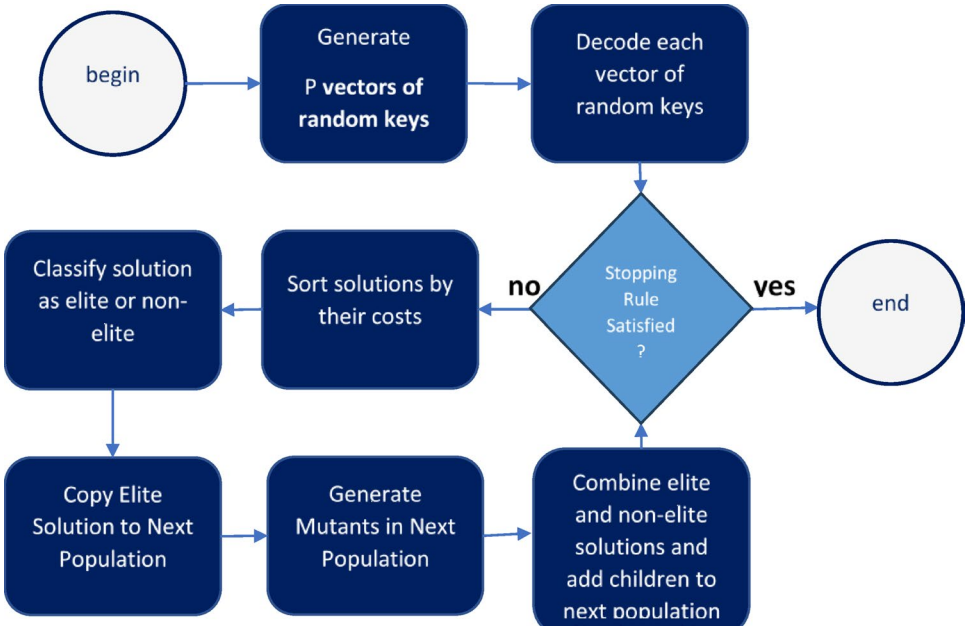

Fig. 12 BRKGA flowchart. Source: research data.

Algorithm 1: BRKGA with type-aware chromosome

Input: $p_{elite}, p_{mutant}, n_{generations}, not_{improve-stop}$, drone cost ($\tau^D$), truck cost ($\tau^D$), SL, SR, E, V_{drone}
Output: best individual

1: Generate Randomly an initial population P
2: **for** $generation = 1 \ldots n_{generations}$ **do**
3: Decode each type of aware chromosome of P with negative signal change if drone tour is better and has conditions necessary to flight and evaluate the fitness
4: Sort chromosome, classify as P_{elite}, $P_{nonelite}$ and getting chromosome with best fitness
5: Randomly generate a new set P_{mutant} of chromosome
6: Generate the offspring set $P_{offspring}$ by using Parametric Uniformed Crossover (PUX), giving random parents from elite population P_{elite} and other parents from non-elite $P_{not\text{-}elite}$
7: $P \rightarrow P_{elite} \cup P_{mutant} \cup P_{offspring}$
8: **if** $not_{improve} < not_{improve-stop}$ **then**
9: break
10: **end if**
11: **end for**

4.4.6 Performance measurement

The algorithm's performance has been evaluated using the GAP measure, which quantifies the relative percentage deviation of a solution. This metric is calculated from the relative deviation between the best solution of algorithm 1 (BS1) and algorithm 2 (BS2), as explained by Eq. 32.

$$GAP_2^1\ (\%) = \left(\frac{BS1 - BS2}{BS2} \right) * 100 \tag{32}$$

Furthermore, the statistical analysis of the computational results will include box plots, a graphical tool that complements numerical data and helps improve understanding of the performance evaluations derived from these results (Talbi 2009). It is advantageous when analyzing cost and time metrics in optimization algorithms. Providing a concise visual summary, it aids in assessing the distribution, central tendency, and variability of these metrics, thereby enhancing the understanding of the optimization algorithm's performance.

5 Computational results

To test the BRKGA algorithm, the TSPlib dataset generated by Reinelt (1991) was used, and the computational results are shown in Table 5. The first column displays the instance serial number. The second column displays the instance name in the format {"name" + "number of points"}. Each line in this dataset can potentially expand the instances addressed. The third column lists the best-known solution (BKS) encountered by Reinelt (1991). Next, statistical measures are used to evaluate the objective function (OF) over 10 runs, including the best value encountered by BRKGA, the average value, standard deviation, and the gap between BKS and the BRKGA results in this study (GAP_{BKS}^{BRKGA}). The final two columns show the best and average execution times in seconds.

Table 5 shows that BRKGA produces good computational results when applied to the TSP problem using the TSPLib dataset. It achieved less than 1% GAP in 26 instances, between 1% and 3% GAP in 8 instances, and between 4% and 10% GAP in 5 instances, resulting in an average GAP of 1.4% across all instances. This is a strong result compared to the BKS of Reinelt (2024).

The average of in column 5 also indicates the consistency of the execution because the average result is close to the best solution within an average standard deviation of 213,3. The best execution time in column 8 varies depending on the instance complexity, ranging from 4 s to 3699 s, with an average of 500s. The average execution time ranges from 4 s to 4463 s, with an overall average of 975s. This demonstrates BRKGA's strong performance in solving the TSP problem.

In Fig. 13, the left side shows the Traveling Salesman Problem (TSP) cost GAP, indicating a minimal deviation from the best-known solution with a mean of 1.4 and

Table 5 Computational results of BRKGA in TSPLib

Nr	Instance	Best Known Solution (BKS)	Objective Function (OF)				Time (s)	
			Best OF	Average OF	Standard deviation	GAP_{BKS}^{BRKGA} %	Best execution time (s)	Average execution time (s)
1	**pr144**	58,537	58,537	58558,0	32,5	**00**	90,0	**90,3**
2	**pr107**	44,303	44,303	44303,0	0,0	**00**	24,0	**32,3**
3	**pr76**	108,159	108,159	108159,0	0,0	**00**	9,0	**11,7**
4	**pr264**	49,135	49,135	49434,7	253,5	**00**	446,0	**663,3**
5	**pr226**	80,369	80,369	80404,3	48,6	**00**	164,0	**335,3**
6	**pr124**	59,030	59,031	59031,0	0,0	**00**	44,0	**50,0**
7	**ts225**	126,643	126,646	128170,7	2156,2	**00**	112,0	**434,7**
8	**pr152**	73,682	73,684	73775,3	64,6	**00**	99,0	**108,7**
9	**berlin52**	7542	7544	7544,0	0,0	**00**	4,0	**4,0**
10	**lin105**	14,379	14,383	14383,0	0,0	**00**	47,0	**52,7**
11	**ch130**	6110	6113	6121,0	5,7	**00**	99,0	**106,7**
12	**rd100**	7910	7918	7935,3	12,3	**0,1**	34,0	**37,7**
13	**pr136**	96,772	96,926	97278,7	268,2	**0,2**	47,0	**58,0**
14	**bier127**	118,282	118,550	118819,3	213,1	**0,2**	78,0	**111,7**
15	**d198**	15,780	15,825	15851,0	36,1	**0,3**	149,0	**339,3**
16	**st70**	675	677	677,3	0,5	**0,3**	9,0	**12,3**
17	**pr299**	48,191	48,334	48758,0	589,8	**0,3**	223,0	**1025,0**
18	**tsp225**	3916	3929	3996,3	48,1	**0,3**	145,0	**311,7**
19	**fl417**	11,861	11,914	11966,3	37,1	**0,4**	1073,0	**2320,3**
20	**ch150**	6528	6559	6559,0	0,0	**0,5**	156,0	**175,7**
21	**p654**	34,643	34,825	35036,0	228,2	**0,5**	1025,0	**4128,3**
22	**rat99**	1211	1219	1219,0	0,0	**0,7**	32,0	**37,3**
23	**eil51**	426	429	429,0	0,0	**0,7**	4,0	**6,7**
24	**u159**	42,080	42,393	42449,3	79,7	**0,7**	64,0	**137,0**
25	**pr439**	107,217	108,047	110244,7	1661,6	**0,8**	515,0	**2221,3**
26	**lin318**	42,029	42,405	42733,7	394,9	**0,9**	673,0	**1344,0**
27	**gil262**	2378	2404	2442,7	29,6	**11**	145,0	**549,3**
28	**a280**	2579	2608	2651,3	59,9	**11**	370,0	**851,7**
29	**rat195**	2323	2355	2400,0	57,4	**1,4**	86,0	**325,3**
30	**pcb442**	50,778	51,580	52070,7	597,0	**1,6**	1938,0	**4022,3**
31	**eil101**	629	643	646,7	2,6	**22**	49,0	**86,7**
32	**eil76**	538	550	550,7	0,5	**22**	15,0	**17,3**
33	**d493**	35,002	35,843	36414,3	422,9	**2,4**	1206,0	**2332,0**
34	**linhp318**	41,345	42,396	43067,7	476,9	**2,5**	580,0	**985,3**
35	**u574**	36,905	38,580	38961,3	278,0	**4,5**	1586,0	**2327,7**
36	**rd400**	15,281	16,061	16106,3	32,7	**5,1**	683,0	**907,0**
37	**u724**	41,910	44,703	44934,7	176,2	**6,7**	3699,0	**4463,3**
38	**rat575**	6773	7225	7289,0	45,3	**6,7**	1437,0	**2910,7**
39	**rat783**	8806	9626	9633,7	10,1	**9,3**	2361,0	**4117,3**
Median/Mean		-	-	-	46,7/213,3	**0,5/1,4**	145,0/500,5	**330,3/975,7**
Min/Max		-	-	-	0,0/2156,2	**0,0/9,3**	4,0/3699,0	**4,0/4463,3**

Source: research data

a median of 0.5. On the right side of Fig. 13, the average execution time is displayed, demonstrating good time performance with a mean of 975 s and a median of 325 s. These computational results emphasize the algorithm's effectiveness in solving the challenges presented by the TSPLIB dataset.

5.1 Small instances – MM, SA, BRKGA

Table 6 shows the computational results for various instances across the mathematical model (MM), Simulated Annealing (SA), and BRKGA. The columns list the instance name and objective function values for each method. The last two columns display the GAP between MM and BRKGA and

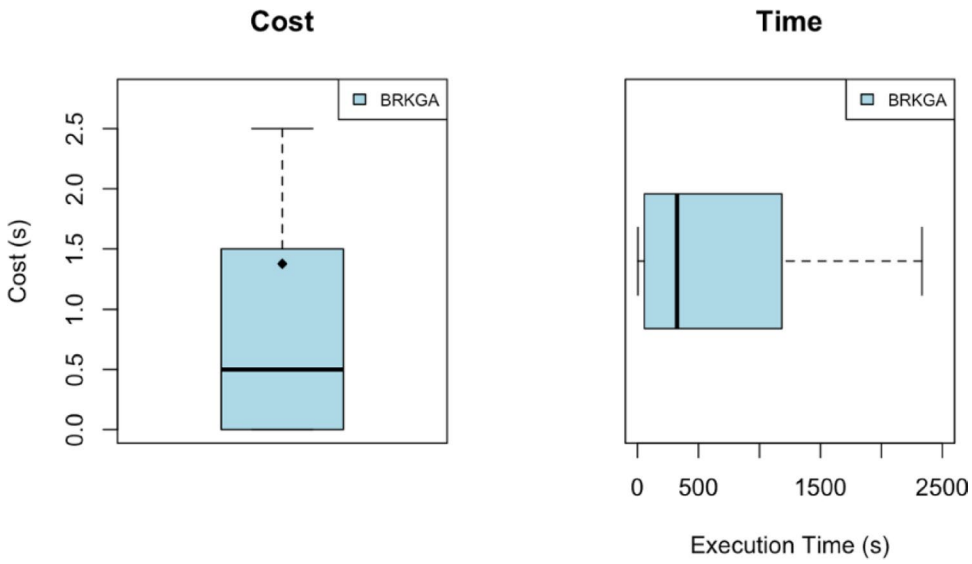

Fig. 13 BRKGA GAP of TSP. Source: research data.

between SA and BRKGA. The previous row presents the average GAP values.

Ten runs of BRKGA were executed and compared to Simulated Annealing by Ponza (2019) and the mathematical model. It was observed that BRKGA outperforms Simulated Annealing in three instances (007.3, 008.4, 010.5) and performs equally well in seven instances (005.1, 005.4, 005.5, 006.1, 006.5, 007.5, 009.1). Moreover, BRKGA demonstrated dominance over SA in 10 of the 20 instances evaluated, underscoring its equal or better performance in half.

In Fig. 14, MM has a mean of 6284.7 and a median of 6352.5, whereas SA shows a mean of 6351.4 and a median of 6503.4. These results indicate that BRKGA's performance is similar to that of SA in smaller instances. However, in the future, drone flights can be further improved to cover more ground and lower costs.

Figure 14 compares execution times for BRKGA, SA, and MM. The first column shows the instance names, while columns 2 to 4 display the execution times (in seconds) for each algorithm across 20 small instances. The last row summarizes the overall performance with the mean, median, minimum, and maximum times for each algorithm. This highlights the strong performance of BRKGA over time, with a mean and median of 0.04s, compared to SA's median of 12.3s and mean of 2.8s, and MM's median of 22.7s and mean of 4562.3s.

As shown in Fig. 14, BRKGA's execution time performs well for small instances. In summary, the application of BRKGA to FSTSP is analyzed using small instances from Ponza (2016). BRKGA outperformed Ponza's (2016) Simulated Annealing in three cases and showed comparable performance in seven cases. Additionally, BRKGA achieved an average execution time of 0.41 s, outperforming both Simulated Annealing (12.34 s) and the Mathematical Model (4,562.32 s).

5.2 Medium and large instances

Table 7 provides a detailed cost comparison for large instances across four optimization algorithms: HTGVNS, HGVNS, SA, and BRKGA, to compare medium and large instances. Instances are categorized by size and sequentially numbered, with each row representing a specific instance. The first column of Table 7 is the best-known solution (BKS) from De Freitas and Penna (2020), followed by the columns HTGVNS, HGVNS, SA, and BRKGA, which show the average cost for each algorithm's optimization on the respective instances. The last four columns display the average GAP of these algorithms, respectively, giving insights into their relative cost performance. The final row of Table 7 presents the average GAP values, offering a consolidated overview of overall cost performance trends.

Regarding cost, HTGVNS achieved an average of −0.01s compared to BKS, while HGVNS reached 0.03s, SA achieved 0.10s, and BRKGA reached 0.18s. Figure 15 shows the HGTVNS, HGVNS, SA, and BRKGA boxplots for medium and large instances. The y-axis represents the cost, and the x-axis indicates each algorithm. The graph displays the data distribution using box plots.

HTGVNS in Fig. 15 has achieved the best computational results, surpassing the best-known solution with a mean of −0.01, followed by HGVNS at 0.03, SA at 0.1, and BRKGA at 0.18. One factor contributing to BRKGA's results is the limited implementation of drone flight.

To compare the optimization algorithm times, Table 7 provides a comprehensive comparison of HTGVNS, HGVNS, and SA BRKGA across 20 medium and large instances, with 50, 100, 150, and 200 points considered. Each row corresponds to a specific instance, categorized by size and a sequential number. The columns HTGVNS, HGVNS, SA,

Table 6 BRKGA FSTSP cost in small instances

			Cost				Time	
Instance	MM Ponza (2016)	SA Ponza (2016)	BRKGA	GAP^{SA}_{MM} (%)	GAP^{BRKGA}_{MM} (%)	BRKGA	SA Ponza (2016)	MM Ponza (2016)
005.1	4456,8	4456,8	4456,8	**0,00**	**0,00**	0,3	5,48	0,98
005.2	3507,1	3507,1	3652,9	**0,00**	4,16	0,3	62,97	0,83
005.3	3275,7	3275,7	3436,8	**0,00**	4,92	0,3	0,01	0,98
005.4	5312,5	5312,5	5312,5	**0,00**	**0,00**	0,2	0,05	0,42
005.5	5510,2	5510,2	5510,2	**0,00**	**0,00**	0,2	16,9	0,64
006.1	7080,9	7080,9	7080,9	**0,00**	**0,00**	0,3	0,01	0,94
006.2	6148,0	6148,0	6426,8	0,00	4,53	0,3	1,3	0,92
006.3	6835,2	6835,2	6895,1	0,00	0,88	0,3	0,01	0,86
006.4	4402,1	4402,1	4619,1	0,00	4,93	0,3	3,65	1,23
006.5	5392,1	5392,1	5392,1	**0,00**	**0,00**	0,3	32,83	1,46
007.1	5533,9	5533,9	5765,5	**0,00**	4,19	0,4	0,46	6,94
007.2	5342,7	5342,7	5382,8	**0,00**	0,75	0,4	1,98	8,54
007.3	7725,9	7783,9	7725,9	0,75	**0,00**	0,3	8,79	2,17
007.4	7610,4	7610,4	7617,1	**0,00**	0,09	0,3	0,4	2,33
007.5	7011,0	7011,0	7011,0	**0,00**	**0,00**	0,4	9,05	7,44
008.1	6709,0	6709,0	6763,1	**0,00**	0,81	0,4	1,07	36,78
008.2	6587,2	6587,2	6746,7	**0,00**	2,42	0,5	0,1	55,44
008.3	5780,1	5780,1	6213,3	**0,00**	7,49	0,5	53,24	47,14
008.4	6505,1	6914,8	6579,9	6,30	**1,15**	0,4	0,1	52,02
008.5	5953,5	5953,5	6336,3	**0,00**	6,43	0,5	7,61	70,61
009.1	7338,8	7338,8	7338,8	**0,00**	**0,00**	0,4	0,76	333,33
009.2	6204,6	6204,6	6240,7	**0,00**	0,58	0,5	39,82	377,46
009.3	7698,1	7698,1	7836,0	**0,00**	1,79	0,4	16,8	418,6
009.4	6817,7	6817,7	6897,0	**0,00**	1,16	0,5	0,06	245,47
009.5	7802,7	7802,7	7904,1	**0,00**	1,30	0,4	6,64	495,74
010.1	5986,7	5986,7	6137,9	**0,00**	2,53	1	1,07	18541,91
010.2	6394,4	6394,4	6654,9	**0,00**	4,07	0,6	37,42	80298,63
010.3	6310,6	6585,4	6599,8	**4,35**	4,58	0,6	56,06	25097,39
010.4	8377,9	8377,9	8503,8	**0,00**	1,50	0,5	1,23	3791,53
010.5	8934,4	8984,2	8934,4	0,56	**0,00**	0,6	4,47	6970,96
MEDIAN/ MEAN	6325,6/ 6284,8	6627,4/ 6415,9	6503,4/ 6351,4	**0,00/ 0,40**	**1,16/ 2,01**	0,40/ 0,41	2,81/ 12,34	22,66/ 4562,32
MIN/ MAX	3275,7/ 8934,4	3275,7/ 8984,2	3436,8/ 8934,4	**0,00/ 0,06**	**0,00/ 0,07**	0,2/ 1,0	0,01/ 62,97	0,42/ 80298,63

Source: research data

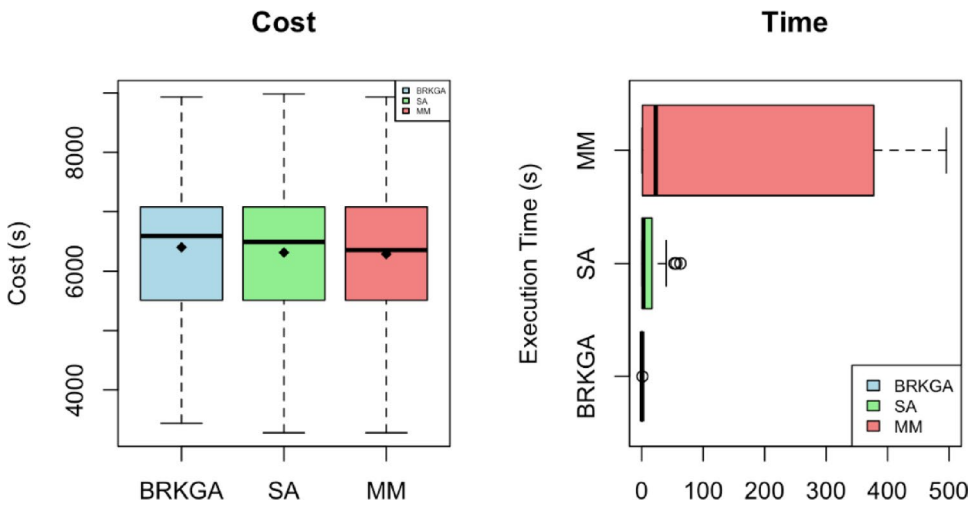

Fig. 14 SA, BRKGA cost in small instance comparison. Source: research data.

Table 7 Cost comparison among HTGVNS, HGVNS, SA, BRKGA

Instances		$\overline{Sol.}$				$\overline{Gap}$ (%)				Time(s)			
	BKS	HTGVNS	HGVNS	SA	BRKGA	HTGVNS	HGVNS	SA	BRKGA	HTGVNS	HGVNS	SA	BRKGA
		De Freitas, et al. (2023)	De Freitas and Penna (2020)	Ponza (2016)	ThisStudy	De Freitas, et al. (2023)	De Freitas and Penna (2020)	Ponza(2016)	ThisStudy	De Freitas et al. (2023)	De Freitas and Penna (2020)	Ponza (2016)	This study
050.1	11506,50	11979,39	11857,02	12518,93	13540,60	0,04	0,03	0,09	0,18	6,98	4,39	213,87	4,50
050.2	10964,30	10984,98	11049,21	12475,14	13749,20	0,00	0,01	0,14	0,25	5,38	4,41	208,36	5,40
050.3	11336,40	11132,43	11336,40	12664,65	13763,70	−0,02	0,00	0,12	0,21	5,14	4,03	191,04	5,30
050.4	10856,40	10692,38	11929,50	12908,18	14117,50	−0,02	0,10	0,19	0,30	5,98	5,02	184,85	4,10
050.5	10846,30	10401,25	11034,30	12164,83	13678,00	−0,04	0,02	0,12	0,26	4,35	4,52	189,86	4,30
100.1	15618,00	15832,94	15623,84	17974,85	19453,20	0,01	0,00	0,15	0,25	15,49	11,94	267,42	28,30
100.2	14899,20	14319,02	15127,50	17342,18	18975,70	−0,04	0,02	0,16	0,27	14,76	9,43	272,35	18,50
100.3	14524,50	14301,39	16074,42	17181,88	19382,70	−0,02	0,11	0,18	0,33	14,30	10,93	265,45	18,50
100.4	15947,30	15604,28	15947,30	18538,03	20025,70	−0,02	0,00	0,16	0,26	13,67	10,34	266,75	22,00
100.5	14948,50	15048,43	15479,22	17407,43	19330,20	0,01	0,04	0,16	0,29	17,29	9,94	312,77	18,60
150.1	19828,10	20042,43	20069,32	22823,38	24395,10	0,01	0,01	0,15	0,23	18,84	20,84	365,04	38,10
150.2	20949,30	21132,30	21390,32	22549,55	23944,00	0,01	0,02	0,08	0,14	27,44	25,05	383,72	44,40
150.3	22633,30	22619,39	23108,49	23114,14	24982,60	0,00	0,02	0,02	0,10	21,39	29,97	379,99	58,20
150.4	20400,70	20248,43	23390,90	22651,00	24207,90	−0,01	0,15	0,11	0,19	27,20	25,39	382,67	61,80
150.5	22435,52	22798,66	23032,05	22807,41	24543,90	0,02	0,03	0,02	0,09	26,78	31,84	384,69	68,00
200.1	25648,33	25700,32	25983,49	26991,21	27157,20	0,00	0,01	0,05	0,06	34,83	30,44	456,74	151,40
200.2	27632,40	25803,29	27985,35	27848,14	28511,80	−0,07	0,01	0,01	0,03	51,39	71,39	452,88	81,00
200.3	26498,33	25193,48	26837,33	27143,78	27927,20	−0,05	0,01	0,02	0,05	57,98	70,38	510,11	73,30
200.4	28247,92	27003,98	28463,95	28503,18	28868,90	−0,04	0,01	0,01	0,02	73,83	87,98	517,44	44,50
200.5	24987,56	25204,44	26357,49	27875,87	28302,10	0,01	0,05	0,12	0,13	53,08	69,38	515,30	67,90
MEDIAN/ MEAN	-	-	-	-	-	**−0,01/−0,01**	**0,02/0,03**	**0,12/0,10**	**0,20/0,18**	**18,06/ 24,80**	**16,39/ 26,88**	**338,90/ 336,06**	**33,20/ 40,90**
MIN/MAX	-	-	-	-	-	**−0,07/0,04**	**0,00/0,15**	**0,01/0,19**	**0,02/0,33**	**4,35/ 73,83**	**4,03/ 87,98**	**184,85/ 517,44**	**4,10/ 151,40**

Source: research data

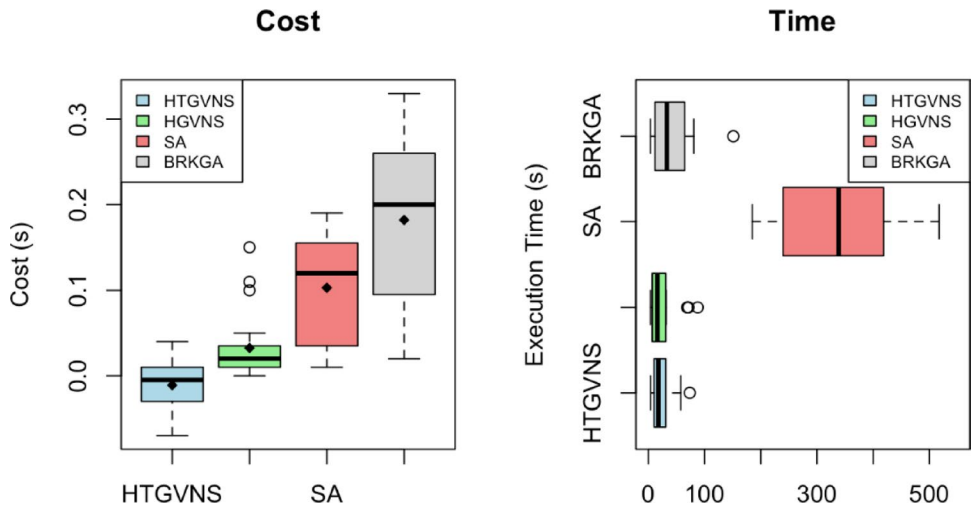

Fig. 15 Performance comparison of HTGVNS, HGVNS, BRKGA, and SA.
Source: research data.

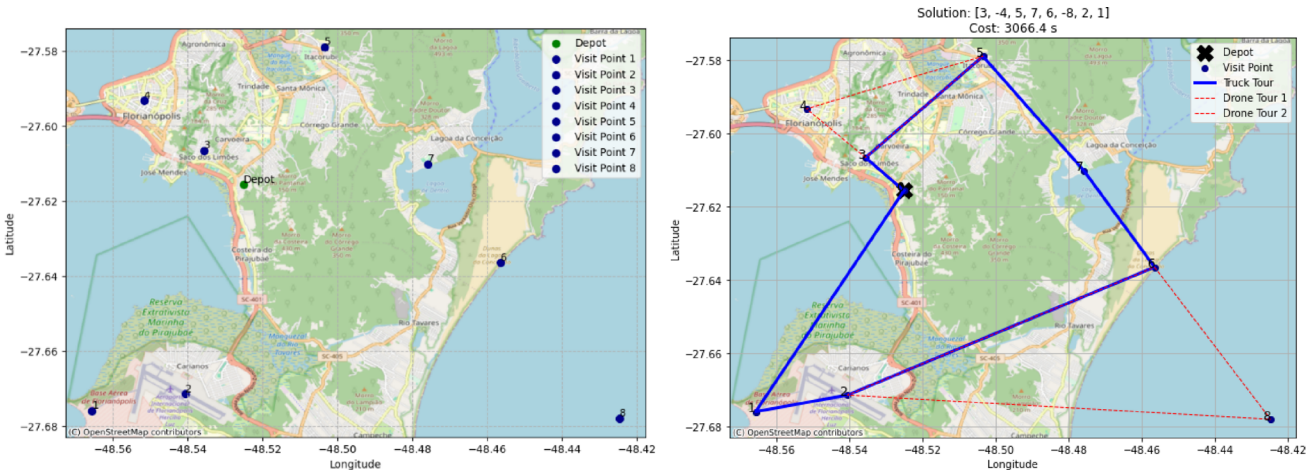

Fig. 16 Florianópolis scenario.
Source: research data.

and BRKGA in Table 7 show the time (in seconds) each algorithm took to optimize the respective instances over 10 runs. The last row indicates the average time for each optimization algorithm. For example, in instance 50.1, HTGVNS took an average of 6.98 s, HGVNS took 4.39 s, SA took 213.87 s, and BRKGA took 4.5 s, respectively. The mean execution times for HTGVNS, HGVNS, SA, and BRKGA across these medium and large instances are 24,80, 26,88, 336,06, and 40,90 s, respectively. This indicates that HTGVNS and HGVNS spend approximately 25 s solving these instances on average. In comparison, BRKGA requires an average of 40.91 s, and SA takes considerably longer at an average of 336.07 s.

The findings in Table 7 are further illustrated with a boxplot, as shown in Fig. 15. The x-axis of the graph represents the algorithms HTGVNS, HGVNS, BRKGA, and SA, while the y-axis indicates the execution time values for Medium and Large Instances. The boxplot includes four different representations, each providing insight into the distribution and variability of execution times across various algorithms and instance sizes.

These findings are further illustrated in Fig. 15, where the boxplot graphic displays the distribution and variability of execution times across the algorithms HTGVNS, HGVNS, BRKGA, and SA for both medium and large instances. HTGVNS shows a mean execution time of 24.8 s and a median of 18.06 s, followed by HGVNS with mean and median times of 26.88 s and 16.39 s, respectively. BRKGA has a mean processing time of 40.90 s and a median of 33.20 s, while SA records a mean of 336.06 s and a median of 338.90 s. This significant variation highlights BRKGA's favorable execution time compared to the state-of-the-art literature, with notably better performance than SA. Each boxplot provides valuable insights into the distribution and variability of execution times, enhancing our understanding of algorithmic performance across different instance sizes.

In summary, when BRKGA is applied to medium and large instances from Ponza (2016), it shows good performance

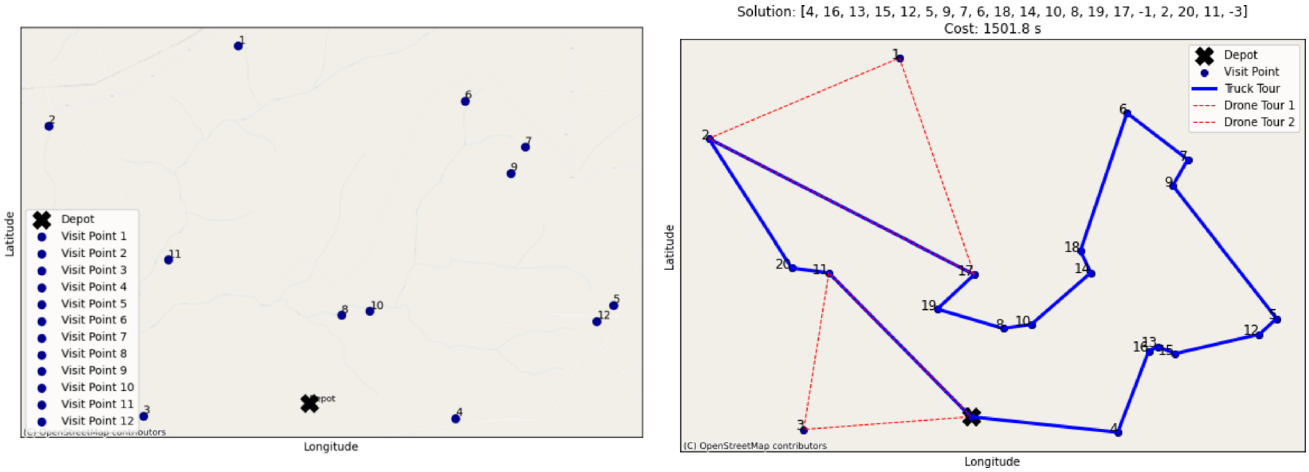

Fig. 17 Army advanced aviation course.
Source: research data.

in terms of time compared to Simulated Annealing. It outperforms SA from Ponza (2016) in execution time, with a median of 33.2 s compared to SA's 338.9 s. However, in terms of solution quality, SA achieved a GAP of 0.10, while BRKGA's GAP is 0.18. Regarding other algorithms, HTGVNS and HGVNS, BRKGA performs close to the median execution times of HTGVNS (18.06 s) and HGVNS (16.39 s). Nonetheless, in solution quality, HTGVNS and HGVNS remain state of the art, with median GAPs of −0.01 and 0.01, respectively, relative to the best-known solution.

5.3 Case 1 – Florianópolis/SC

Figure 16 is chosen to provide a clear view of applying optimization algorithms in a real-world civilian setting. It depicts a scenario for studying airspace in Brazil by the Laboratory Concept from ITA. This figure shows a map with eight distinct blue visit points and one red depot point, which serves as the starting and ending point for both truck and drone routes, as indicated in the legend. The y-axis represents latitude, while the x-axis shows longitude, providing a spatial view of the geographical coordinates.

Using OpenStreetMap as the basemap, this visualization was created with the GeoPandas library in Python, a flexible tool for spatial data analysis and visualization. This approach thoroughly examines the optimized routes and their spatial distribution within Brazilian airspace.

Running the BRKGA optimization algorithm, the route solution [3, −4, 5, 7, 6, −8, 2, 1] as shown in Fig. 16 is reached, costing 3066.4 s. This saves 102.6 s compared to drone tour 1 [3, 4, 5] and for drone tour 2 [6, 8, 2], highlighting the benefit of visiting a location inaccessible to the truck.

Compared to alternative solutions, the route obtained [3, −4, 5, 7, 6, −8, 2, 1] stands out significantly. For example, if the point in water eight is disregarded, the simple solution relying solely on trucks [4, 3, 1, 6, 7, 2, 5] incurs a total cost of 4316.7 s, which is 40.7% higher than the optimized route. These comparisons highlight the advantages of using BRKGA in the FSTSP, emphasizing its efficiency and effectiveness in minimizing travel costs while maximizing operational performance. In short, running the BRKGA optimization algorithm on the Florianópolis case study yielded the route solution [3, −4, 5, 7, 6, −8, 2, 1] with a cost of 3066.4 s. This results in a saving of 102.6 s for drone tour 1 [3, 4, 5] and offers the benefit of visiting a location inaccessible to the truck in drone tour 2 [6, 8, 2].

5.4 Case 2 – Army advanced aviation course

Figure 17 illustrates a scenario involving the Advanced Aviation Course of the Brazilian Army, including 13 strategic points such as the depot, to show how optimization algorithms can be used in a military context. These points include maintenance facilities, observation outposts, logistical distribution centers, operational bases, and air operational bases. This scenario highlights the complexities of military operations and provides an army setting for applying optimization algorithms.

Running the BRKGA optimization algorithm, the route solution [4, 16, 13, 15, 12, 5, 9, 7, 6, 18, 14, 10, 8, 19, 17, −1, 2, 20, 11, −3] is encountered with a cost of 1501.8 s, as shown in Fig. 17. This represents a saving of 31.1 s in drone tour 1 [17,1,2] and 5.4 s in drone tour 2 [11,3,0]. The advantages of using drones could still be improved by implementing drone flight while the truck is moving, considering more realistic data for the truck's distance and velocity. Nonetheless, the drone can be effectively used in military operations in strategic and sensitive areas.

Compared to alternative solutions, the obtained route [4, 16, 13, 15, 12, 5, 9, 7, 6, 18, 14, 10, 8, 19, 17, −1, 2, 20, 11, −3] stands out notably. For example, the naive solution relying solely on trucks [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20] results in a total cost of 3224.1, representing a significant 114.6% improvement with the optimized route. Additionally, when compared to another solution [8, 10, 4, 16, 13, 15, 5, 12, 9, 6, 7, 18, 14, 10, 8, 19, 17, 2, 1, 20, 11, 3] with a total cost of 1797.8, the BRKGA solution shows a 19.7% improvement. These comparisons highlight the benefits of using BRKGA in the FSTSP, demonstrating its efficiency and effectiveness in reducing travel costs and improving operational performance.

To sum up, the route solution [4, 16, 13, 15, 12, 5, 9, 7, 6, 18, 14, 10, 8, 19, 17, −1, 2, 20, 11, −3] in the Advanced Aviation Course scenario was achieved with a cost of 1501.8 s, resulting in a savings of 31.1 s for drone tour 1 [17,1,2] and 5.4 s for drone tour 2 [11,3,0]. The benefits of using drones can still be enhanced by deploying drone flights while the truck is moving and by considering more realistic distances and velocities for the truck. Additionally, drones can be utilized in strategic and sensitive areas during military operations.

This study makes valuable contributions by tackling complex challenges in route optimization. It investigates the use of a previously untested metaheuristic for this problem, introduces new methods for route optimization with drone assistance, and offers innovative insights through a BRKGA encoder that uses real numbers. This innovation enables the integration of trucks and drones within the same data structure, a conceptual advancement not found in existing literature except for a preprint by Mahmoudinazlou and Kwon (2023). As a result, it establishes a foundation for developing more efficient heuristic algorithms in truck and drone routing optimization, as noted by researchers such as Chunget al. (2020).

6 Conclusion

In this study, the Biased Random-Key Genetic Algorithm (BRKGA) proved to be a promising approach for the Flying Sidekick Traveling Salesman Problem (FSTSP), emphasizing its potential uses in both civilian and military settings for hybrid truck and drone delivery systems. The goal of applying BRKGA to the Traveling Salesman Problem (TSP), a routing challenge that involves only trucks and no drones, was accomplished, as solving this problem efficiently provides a solid foundation for tackling the FSTSP effectively. The algorithm demonstrated encouraging computational results when tested on the TSP using the TSPLib Dataset, achieving less than 1% GAP in 26 instances, between 1% and 3% GAP in 8 instances, and between 4% and 10% GAP in 5 instances, indicating good solution quality. In terms of execution time, it also performed well, with durations ranging from 4 s to 4463 s, depending on the size and complexity of each instance.

The main goal of implementing BRKGA in FSTSP, a variation of TSP with a drone synchronized to the truck, has been achieved. In a small dataset from the literature, BRKGA outperformed three instances and performed equally well in seven instances in terms of solution quality. Regarding execution time, BRKGA showed an average of 0.41 s, faster than both Simulated Annealing (12.34 s) and the Mathematical Model (4562.32 s). When applied to medium and large instances from Ponza (2016), BRKGA demonstrated better performance in execution time compared to Simulated Annealing (SA). It achieved an average of 40.9 s, beating SA's 336.06 s. Although BRKGA is quicker, it slightly trails SA in solution quality, with a GAP of 0.18 versus SA's 0.10. Additionally, BRKGA's time performance is similar to HTGVNS and HGVNS, with average times close to 18.06 s and 16.39 s, respectively. However, HTGVNS and HGVNS remain the top methods for solution quality, with median GAPs of −0.01 and 0.01, respectively.

In the pursuit of real-world applications, implementing the BRKGA optimization algorithm in the Florianópolis case study led to significant cost savings and operational benefits. Compared to alternative solutions, the optimized route achieved an impressive 40.7% reduction in total costs, demonstrating the efficiency and effectiveness of BRKGA in reducing travel expenses while improving operational performance in the FSTSP scenario. Notably, the optimized route saved 102.6 s for drone tour one and allowed access to previously unreachable locations for the truck in drone tour 2. These findings highlight the tangible advantages of utilizing BRKGA in practical civilian scenarios.

In a military context, the BRKGA solution in the Advanced Aviation Course scenario showed significant performance improvements, with a 114.6% increase over a naive solution that relied only on trucks and a 19.7% increase over another alternative solution. These comparisons highlight the efficiency and effectiveness of using BRKGA to reduce travel costs while maximizing operational performance in the FSTSP. Additionally, in military operations, drones can be strategically deployed in sensitive areas, further improving operational capabilities, potentially lowering route deviations and execution times, and boosting military readiness and responsiveness by integrating drones as support in logistical missions. This underscores the importance of ongoing development efforts in optimization algorithms like BRKGA to strengthen drone-assisted military operations.

This study benefits society by promoting innovation, optimizing resource allocation, and encouraging the adoption of new solutions across various industries. Especially in the logistics and surveillance sectors, where drones play key roles, this research can significantly boost operational efficiency and resource management. Combining trucks and drones can speed up delivery, cut down carbon emissions, and lower logistical costs for businesses.

In the field of air transportation, developing route optimization strategies is vital in a sector where drones are increasingly used alongside traditional aircraft. This approach offers potential improvements in efficiency and safety. Additionally, it encourages innovation in the ongoing development of the aviation industry, supporting the seamless integration of drone technology into the current air transport system. Through this structured approach, the study aims to improve understanding and solution approaches for routing problems, especially in drone integration scenarios, thus laying the foundation for practical applications and theoretical progress in this area.

Looking ahead, future research can explore hybrid algorithmic strategies and multi-objective extensions that balance cost, time, energy use, and operational risk. Adapting the algorithm to a dynamic and stochastic environment will be crucial for tackling real-time logistics challenges. Additionally, integrating with real-time decision-support systems and applying to regulatory constraints, humanitarian efforts, and emergencies could open new avenues for research. These approaches enhance BRKGA and broaden its applicability and impact in real-world scenarios where truck-drone collaboration offers a disruptive logistical advantage.

Funding The Article Processing Charge (APC) for the publication of this research was funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (ROR identifier: 00x0ma614).

Data availability Data will be made available on request.

Declarations

Competing interests The authors declare no competing interests.

Ethical approval Not applicable.

References

Agatz N, Bouman P, Schmidt M (2018) Optimization approaches for the traveling salesman problem with drone. Transp Sci 52(4):965–981. https://doi.org/10.1287/trsc.2017.0791

Akşit B, Gürcan GÖ, Sariçiçek İ (2024) Flying sidekick traveling salesman problem in truck–drone team logistics with energy issues. Transp Res Rec. https://doi.org/10.1177/03611981241243076

Baldisseri A, Siragua C, Seghezzi A, Mangiaracina R (2022) Truck-based drone delivery system: an economic and environmental assessment. Transp Res Part D Transp Environ 107:103296. https://doi.org/10.1016/j.trd.2022.103296

Benarbia T, Kyamakya K (2022) A literature review of drone-based package delivery logistics systems and their implementation feasibility. Sustain Switz 14(1):369. https://doi.org/10.3390/su14010360

Blum C, Puchinger J, Raidl GR, Roli A (2011) Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 11(6):4135–4151. https://doi.org/10.1016/j.asoc.2011.02.032

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35(3):268–308. https://doi.org/10.1145/937503.937505

Boccia M, Mancuso A, Masone A, Murino T, Sterle C (2024) New features for customer classification in the flying sidekick traveling salesman problem. Expert Syst Appl 235:121406. https://doi.org/10.1016/j.eswa.2023.123106

Boccia M, Masone A, Sforza A, Sterle C (2021) A column-and-row generation approach for the flying sidekick travelling salesman problem. Transportation Research Part C: Emerging Technologies 124:102913. https://doi.org/10.1016/j.trc.2020.102913

Boussaïd I, Lepagnot J, Siarry P (2013) A survey on optimization metaheuristics. Inf Sci 237:82–117. https://doi.org/10.1016/j.ins.2013.02.041

Chung SH, Sah B, Lee J (2020) Optimization for drone and drone-truck combined operations: a review of the state of the art and future directions. Comput Oper Res 123:105036. https://doi.org/10.1016/j.cor.2020.105004

Coutinho WP, Battarra M, Fliege J (2018) The unmanned aerial vehicle routing and trajectory optimization problem, a taxonomic review. Comput Ind Eng 120:116–128. https://doi.org/10.1016/j.cie.2018.04.037

De Freitas JC, Penna PHV (2018) A randomized variable neighborhood descent heuristic to solve the flying sidekick traveling salesman problem. Electron Notes Discret Math 66:95–102. https://doi.org/10.1016/j.endm.2018.03.013

De Freitas JC, Penna PHV (2020) A variable neighborhood search for flying sidekick traveling salesman problem. Int Trans Oper Res 27(1):267–290

De Freitas JC, Penna PHV, Toffolo T (2023) Exact and heuristic approaches to truck-drone delivery problem. EURO J Transp Logist 12:100094

Dell'Amico M, Montemanni R, Novellani S (2021) Algorithms based on branch and bound for the flying sidekick traveling salesman problem. Omega 104:102493. https://doi.org/10.1016/j.omega.2021.102493

Elshaer R, Awad H (2020) A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. Comput Ind Eng 140:106242

Gonçalves JF, Resende MGC (2011) Biased random-key genetic algorithms for combinatorial optimization. J Heuristics 17(5):487–525. https://doi.org/10.1007/s10732-010-9143-1

González-R P, Canca D, Andrade J, Calle M, Leóón Blanco J (2020) Truck-drone team logistics: a heuristic approach to multi-drop route planning. Transp Res Part C Emerg Technol 114:657–680. https://doi.org/10.1016/j.trc.2020.02.030

Ha QM, Deville Y, Pham QD, Hà MH (2018) On the min-cost traveling salesman problem with drone. Transp Res Part C-Emerg Technol 86:597–621. https://doi.org/10.1016/j.trc.2017.11.015

Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge

Khoufi I, Laouiti A, Adjih C (2019) A survey of recent extended variants of the traveling salesman and vehicle routing problems for unmanned aerial vehicles. Drones 3(3):66. https://doi.org/10.3390/drones3030066

Macrina G, Pugliese LDP, Guerriero F (2020) Drone-aided routing: a literature review. Transp Res C 120:102762. https://doi.org/10.1016/j.trc.2020.102762

Mahmoudinazlou S, Kwon C (2023) A hybrid genetic algorithm with type-aware chromosomes for traveling salesman problems with drone. vol. 2, pp 1-40. ArXiv Preprint. http://arxiv.org/abs/2303.00614

Moshref-Javadi M, Winkenbach M (2021) Applications and research avenues for drone-based models in logistics: a classification and review. Expert Syst Appl 177:114854. https://doi.org/10.1016/j.eswa.2021.114854

Murray CC, Chu AG (2015) The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. Transp Res Part C-Emerg Technol 54:86–109. https://doi.org/10.1016/j.trc.2015.03.005

Ojeda Rios BH, Xavier EC, Miyazawa FK, Amorim P (2021) Recent dynamic vehicle routing problems: a survey. Comput Ind Eng 160:107567

Otto A, Agatz N, Campbell J, Golden B, Pesch E (2018) Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey. Networks 72(4):411–458. https://doi.org/10.1002/net.21818

Peres F, Castelli M (2021) Combinatorial optimization problems and metaheuristics: review, challenges, design, and development. Appl Sci 11(14):6449. https://doi.org/10.3390/app11146449

Ponza A (2016) Optimization of drone-assisted parcel delivery. Master's dissertation, Università Degli Studi di Padova

Raj R, Murray C (2020) The multiple flying sidekicks traveling salesman problem with variable drone speeds. Transp Res C Emerg Technol 120:102813. https://doi.org/10.1016/j.trc.2020.102813

Reinelt G (1991) TSPLIB—a traveling salesman problem library. ORSA J Comput 3(4):376–384. https://doi.org/10.1287/ijoc.3.4.376

Reinelt G (2024) Optimal solutions for symmetric TSPs. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/. Accessed 1 Feb 2024

Rojas Viloria D, Solano-Charris EL, Muñoz-Villamizar A, Montoya-Torres JR (2021) Unmanned aerial vehicles/drones in vehicle routing problems: a literature review. Intl Trans Op Res 28:1626–1657. https://doi.org/10.1111/itor.12783

Talbi EG (2009) Metaheuristics: from design to implementation. John Wiley & Sons, Inc., Hoboken

Tan SY, Yeh WC (2021) The vehicle routing problem: state-of-the-art classification and review. Appl Sci (Switzerland) 11(21):10291. https://doi.org/10.3390/app112110291

Teimoury E, Rashid R (2024) A hybrid variable neighborhood search heuristic for the sustainable time-dependent truck-drone routing problem with rendezvous locations. J Heuristics. https://doi.org/10.1007/s10732-023-09497-5

Wang Z, Sheu J-B (2019) Vehicle routing problem with drones. Transp Res Part B Methodologic 122:350–364. https://doi.org/10.1016/j.trb.2019.03.005

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893

Yu VF, Lin SW, Jodiawan P, Lai YC (2023) Solving the flying sidekick traveling salesman problem by a simulated annealing heuristic. Mathematics 11(20):4305. https://doi.org/10.3390/math11204305